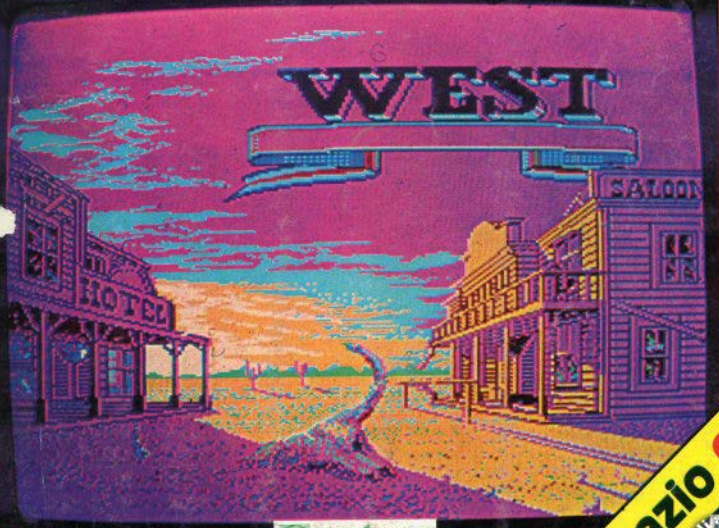
sinclair COMPUTER

mensile per gli utenti dei computer Sinclair

#14 Lire 3000 giugno 1985

Microdrives random

Arrow of death



Sinclair Computer - anno 2 n. 14 - giugno 1985 - L. 3.000 - sped. abb. post. gr. II/70 - Distrib. Ma.Fe.

Systems

Spazio QL

ZX 19

EDIZIONE ITALIANA DI SINCLAIR COMPUTER

MAGAZINE ZX SPECTRUM

anno 2 - n. 4 - Aprile 1985 - Lire 6.500

Parolatore
Oca
Debug
Caratteri

"Il lungo ritorno" no 4

S systems

sommario

05 - Sinclairamente vostro/la posta
07 - Precisazioni e perfezionamenti
08 - Posta adventures
10 - Libri/recensioni
12 - Adventures: ARROW OF DEATH/II parte
(Giuliano Boschi)
15 - Didattica: ASSEMBLY
(Gianluca Carri)
18 - Accesso casuale ai microdrives/II parte
(Damiano Bolla & Renzo Zonin)
21 - SOFTWARE:
Polinomi (G.Alcini & S.Salsano)
Grafici polari (S.Pizzeghello)
Integrali doppi (P.Paccagnella)
LIST control (G.Corbelli)
23 - Bubble-sort in ling.macchina
(Luigi Callegari)
24 - SOFTWARE:
Boxe (A.Ricciuti)
26 - Sinclairparade/la classifica
27 - I LISTATI
37 - Listati QL
43 - I simulatori di volo
(Luigi Callegari)
44 - Un tasto per volta
SPAZIO QL
(Marco Mussini & Roberto Previtera)
46 - Le variabili di sistema
46 - Il linguaggio macchina
48 - Il Superbasic
53 - Un po' di Pascal
(Monica Fumagalli)
58 - Videogames/recensioni
(Luigi Callegari)
60 - Sinclaireclame/piccoli annunci
63 - Un tasto per volta (seguito)

In copertina: "WEST", adventure per il QL.

sinclair COMPUTER

REDAZIONE
Mauro Soldavini, Fabio Berno, Marco De Martino

SEGRETERIA DI REDAZIONE
Maura Ceccaroli, Piera Perin

COLLABORATORI
Paolo Beneventi, Marco Bertani, Carlo Bolchini, Damiano Bolla, Giuliano Boschi, Luigi Callegari, Gianluca Carri, Valerio Cipolla, Paolo Dray, Fabrizio Ferraro, Monica Fumagalli, Guido Grassi, Giovanni Mellina, Marco Mussini, Roberto Previtera, Renzo Zonin.

GRAFICA E IMPAGINAZIONE
Cristiana Goglio

DIFFUSIONE E ABBONAMENTI
Marina Vantini

DIREZIONE, REDAZIONE
Viale Famagosta 75 - 20142 Milano - Tel. (02) 8467348/9/40

PUBBLICITÀ
Milano: Mirco Croce (coordinatore), Giuseppe Porzani, Michela Prandini, Giorgio Ruffoni, Claudio Tidone, Villa Claudio
Segreteria: Liliana Degiorgi
— V.le Famagosta 75, 20142 Milano - tel. (02) 8467348/9/40
Roma: Spazionuovo di R. De Marinis via P. Foscari 70, 00139 Roma tel. (06) 8109679

FOTOCOMPOSIZIONE
Fotocomposizione LM (Brescia)

STAMPA
La Litografica S.r.l. (Busto Arsizio)

DISTRIBUZIONE
Messaggerie Periodici S.p.A.
via G. Carcano 32, Milano
Spedizione in abb. Post. GR. III/70

SYSTEMS EDITORIALE s.r.l.
(Registro Nazionale Stampa n. 01500 vol. 15 fg. 793)
Direttore responsabile: Agostina Ronchetti
Autorizz. Trib. di Mi n. 255/12.11.1983

Una copia L. 3.000 (Arretrati L. 6.000)
Abbonamento annuo (11 numeri) L. 28.000 (estero il doppio). I versamenti e le richieste di arretrati vanno indirizzati a: Sinclair Computer, V.le Famagosta 75, 20142 Milano, mediante emissione di assegno bancario o versamento sul c/c postale n. 30426209. Per i cambi di indirizzo indicare, unitamente al nuovo, anche l'indirizzo precedente, allegando L. 500 in francobolli.

Smau: il giro del mondo in 91.000 metri quadrati

Smau: chi lo visita farà un entusiasmante giro del mondo in 91.000 mq. Qui infatti troverà tutte le novità dei più importanti produttori mondiali. Qui troverà esperti capaci di consigliare le soluzioni più aderenti al futuro dell'azienda e dell'organizzazione del lavoro. Troverà la 18ª edizione del Premio Smau Industrial Design; troverà Convegni e Seminari: troverà lo Spazio Giovani. Troverà il mondo intero: tutto racchiuso in 91.000 metri quadrati.

smau

22º Salone Internazionale per l'Ufficio: sistemi per l'informatica, la telematica, le comunicazioni, macchine, arredamento per l'ufficio

ENTE GESTIONE MOSTRE COMUFFICIO

Quartiere Fiera Milano
19-24 Settembre 1985

Contemporaneamente, 3ª EIMU, Esposizione Internazionale Mobili Ufficio

sinclair amente vostro

RAM fantasma

Ho uno Spectrum 16k. Le locazioni da 32768 in su hanno valore 255 e sono inalterabili: perchè? Provando RANDOMIZE USR x, con x tra 1216 e 1329... che cosa succede? *(F. Giuliano - Catania)*

Le locazioni da 32768 in su, nel 16k, non esistono! Con i 16 kilobytes di ROM, uguali per tutti gli Spectrum, si arriva alla locazione 16383; i 16k di RAM occupano le locazioni da 16384 a 32767. In queste sono compresi: la memoria di schermo, le variabili di sistema, l'area basic, gli stack, gli UDG (quindi nella versione 16k, in realtà, si hanno a disposizione per i programmi basic circa 9k).

Il computer risponde 255 perchè la routine di PEEK, nel sistema operativo, prevede argomenti fino a 65535 (perciò non va in errore) e, non trovando la locazione, ritorna un valore di default.

Le locazioni citate fanno parte delle routines di gestione del registratore: quello che si ottiene è un avvio (fittizio) di un'operazione di SAVE.

Spostare il basic

Come si fa a spostare la locazione di inizio del basic nello Spectrum? *(B. Sforzini - Cervia RA)*

L'operazione è possibile simulando la presenza di una "microdrive map area", a condizione che non si utilizzino microdrives e si operi in linguaggio macchina (occorre un minimo di pratica di assembly); si devono eseguire le seguenti istruzioni:

```
LD HL, (CHANS)
DEC HL
LD BC,n
CALL 5717d
```

in cui CHANS è la variabile di sistema che porta questo nome e *n* il numero di bytes che si vogliono liberare; al rientro dalla CALL, le locazioni da 23734 a 23734 + *n*-1 restano utilizzabili. Il metodo è suggerito dal collaboratore Giovanni Mellina in un libro sul sistema operativo dello Spectrum di imminente uscita nella nostra collana "I libri di System".

Confrontate anche, in questo numero di SC, le informazioni sulla microdrive map area nell'articolo "Accesso random ai mdrives".

Comunicazione tra... padri e figli

È possibile collegare uno ZX81 a uno Spectrum? *(Ivan Cassani - Milano)*

In teoria, i metodi esistono; la via di comunicazione più usata nel collegamento tra due computer è probabilmente la porta seriale, ma sfortunatamente per lo ZX81 non ne sono mai giunte in Italia.

Sappiamo invece dell'esistenza di un software per Spectrum che lo mette in grado di caricare programmi dalle cassette ZX81, ma anche qui c'è una limitazione: l'operazione è a senso unico (dal più vecchio al più giovane).

Monitor

Come si collega lo Spectrum in modo monitor a uno zoccolo SCART? *(vari lettori)*

Si deve collegare la 'massa' dello Spectrum al piedino 17 (Signal Ground) e la linea del segnale composito al piedino 20, bypassando ovviamente il modulatore. Per mettere il televisore in modo monitor occorre poi consultare le istruzioni dell'apparecchio: solitamente vi si accede superando l'ultimo canale, con il telecomando, oppure con un tasto apposito.

Warning buzz

A volte, inserendo attributi di colore o "inverse", lo Spectrum dà un beep e si blocca. *(Sergio Tassi - Firenze)*

Le sequenze di attributi di stampa inserite direttamente nelle stringhe possono creare questi problemi in fase di editing: niente di preoccupante, ma esiste il rischio che il blocco permanga, costringendo a spegnere il computer, perdendo il programma già scritto. È meglio

usare i comandi basic espliciti (PAPER, INK, FLASH, etc.), soprattutto se il programma deve essere portato su carta: gli attributi inseriti nelle stringhe diventano invisibili e possono creare difficoltà a una stampante non Sinclair-dedicata.

C'è incompatibilità tra un dato immesso da un joystick programmabile e quello dalla tastiera? *(Sebastiana Grasso - Roma)*

La domanda non è molto chiara: i dati immessi via joystick o via tastiera, quando giungono alla ULA, sono in ogni caso "dati", cioè numeri. Nel caso del joystick occorre conoscere l'indirizzo della porta di ingresso e quali valori corrispondono alle diverse posizioni: saputo questo, qualsiasi elaborazione è possibile, con l'evidente limitazione che i dati non possono essere più di 5 (o 9).

Le routines di trasferimento

Come si comandano LOAD / SAVE/VERIFY in linguaggio macchina? *(P. Ebner - Roma)*

Alla locazione 1218 della ROM si trova la routine di SAVE. Preparato l'header, come si caricano i registri? *(G. Benintende - Leonforte EN)*

Il MERGE da microdrive non funziona se nel programma c'è l'autostart. Si può aggirare l'ostacolo? *(vari lettori)*

Come si registra e ricarica un programma senza header? Si possono ancora usare VERIFY e MERGE? *(vari lettori)*

Vista la quantità di richieste, il salvataggio e il caricamento di programmi con e senza header, chiamando direttamente le routines apposite del sistema operativo, e il loro trasferimento su microdrive, saranno oggetto di un prossimo articolo, che cercherà di trattare in una sola volta i diversi casi.

Sulla ROM e il sistema operativo

Come utilizzare le routines PLOT, CIRCLE, PRINT AT, RND con una chiamata al I/m? *(Pierluigi Fersini - Taranto)*

Qual è il modo per estrarre numeri random in ling. macchina? Qual è l'indirizzo della routine di SAVE nella ROM, e come devono essere caricati i registri? *(Fabrizio B. Cordero - Baldissero C. TO)*

L'utilizzo delle routines grafiche e di gestione video richiede buona conoscenza del linguaggio macchina e del sistema operativo; un discorso esauriente richiederebbe l'analisi delle routines, e CIRCLE/DRAW sono piuttosto lunghe. Anche a questi lettori consigliamo perciò una bibliografia: i volumi sull'assembly Spectrum della McGraw-Hill, recensiti su SC n. 12.

La routine RND si trova nella ROM da 9725 a 9765, ma non può essere semplicemente chiamata, poichè non ha un'istruzione di ritorno; per utilizzarla la soluzione migliore è quella di ricopiarla nella RAM e chiamarla poi come subroutine. Gli stessi libri offriranno maggiori dettagli.

Per SAVE, cfr. risposta precedente.

Stampanti

La mia Alphacom 32 stampa caratteri più sbiaditi a sinistra. *(Lorenzo Lugli - Pesaro)*

Il difetto (peraltro poco evidente) non è facilmente eliminabile. Un primo intervento può essere il controllo dell'allineamento dei "punti caldi" che provocano l'annerimento della carta termosensibile (si tratta di 16 punti): dove la stampa è sbiadita potrebbero essere più distanti dalla carta. Altre verifiche possono essere fatte solo da un tecnico specializzato.

Come si può ottenere la stampa di caratteri speciali su una stampante a 80 colonne, all'interno di testi prodotti con un word-processor? *(Isabella Pagano - Catania)*

Con il word processor non è possibile (nè sarebbe pratico) eseguire hard-copy del video; per avere caratteri speciali all'interno di un testo è necessario che la stampante disponga dell'opzione "caratteri programmabili", eventualmente con buffer di memoria (tipo Epson FX-80). Le modalità d'uso sono illustrate nel manuale di ciascuna stampante.

Che cos'è il CP/M? *(Sergio Doria - Milano)*

Il CP/M *(Control Program for Microcomputers)*, marchio registrato della Digital Research) non è un tipo di basic, ma un DOS (Disk Operating System, sistema operativo per dischi) per computer che usino i microprocessori Intel 8080 o Zilog Z80. La letteratura sul CP/M è ampia, tutte le migliori collane di informatica hanno almeno un volume che ne parla. Allo Spectrum è collegabile l'unità a dischi "Piccolo Gigante" (v. SC n. 11 pag. 49), prodotta dalla DATA B. di Milano, che mette a disposizione il sistema CP/M dialogando con il computer (visto in questo caso come un terminale) attraverso la porta RS232 dell'interface 1.

Risposte brevi

(Massimo Albini - Milano) Per far eseguire i comandi BORDER e PAPER già al primo ciclo di programma, basta inserire un "clear screen" (CLS) subito dopo di essi.

(Paolo D'Ambrosi - Trieste) L'ipotesi è giusta: se il computer non è stato manomesso, l'inconveniente dipende probabilmente dalla membrana; un centro di assistenza ti saprà dare una risposta più precisa di noi, che non vediamo il computer.

(Andrea Gilli - Cremona) Il corso di assembly per lo Spectrum è iniziato sul n. 04 ed è proseguito per tutti i numeri, con la sola eccezione del 09; sul n. 12 trovi un indice analitico completo dei primi dieci numeri di SC.

(Alberto Roncallo - Genova) Ti consigliamo di leggere la rubrica "Un tasto alla volta", sul n. 12 (pag. 56), relativamente a "POKE": abbiamo l'impressione che tu abbia le idee un po' confuse in merito a questa istruzione basic.

(Ernesto Cappello - Ragusa) Si deve rivolgere alla rivista che ha pubblicato quell'articolo, e non a noi.

Perfezionamenti & precisazioni

TotoSeikosha

Il programma "TotoSeikosha" pubblicato alle pag. 23/33 del n. 12 necessita di una piccola messa a punto per un migliore funzionamento: le linee 2161 e 2167 vanno inserite ex-novo, la 2162 e la 2166 vengono modificate come di seguito.

```
2161 FOR m=0 TO 3: IF m+i>col TH
EN  GO TO 2167
2162 LET j$=c$(i+m): FOR x=1 TO
13

2166 GO SUB 2140: NEXT x
2167 NEXT m
```

Ho fatto 13?

Anche questo Software è apaprso sul n. 12. Nel caso si vogliano controllare più di 20 colonne, occorre inserire le linee 87 e 3027, e modificare le altre qui listate, già esistenti nel programma.

```
  87 DIM m(nc)
3025  FOR z=1 TO nc STEP 20:-FOR
 k=1 TO 20: IF (z+k-1)<=nc THEN
 LET pn=0: FOR j=1  TO 13: LET d
=(d$(j)=c$(z+k-1,j)): GO TO 3030
3027 GO TO 3055
3030 PRINT  INVERSE d;AT 1+j,k+1
0;c$(z+k-1,j): LET pn=pn+d: NEXT
 j
3040 LET m(z+k-1)=pn: IF pn>m1 T
HEN  LET m1=pn
3050 NEXT k: PRINT #0;"premere u
n tasto per le altre   colonne
": PAUSE 0: FOR i=1 TO 13: PRINT
 AT i+1,11;" [ 20 spazi ]
": NEXT i: NEXT z
```

QL - mastermind

Qualche piccolo problema per il "mastermind" destinato al QL (n. 13 pag. 48): nonostante l'assicurazione che fosse libero da errori, il santo protettore dei bug ci ha messo la zampa. Alla linea 350 anzichè CHR$ (0) inserite CHR$ (1); alla 480, invece di CHR$ (0) scrivete CHR$ (2).

Caricamento di hex

Il programma "Istogrammi" (n. 12 pag. 25/34) non era chiaro in merito al caricamento dei codici per la stampa su 64 colonne. Ecco un esempio di come fare: digitate

```
10 CLEAR 64255
20 FOR x = 64256 TO 65055
30 INPUT q$
40 LET q = CODE q$
50 LET q = q-48-7*(q> 64)
60 POKE x,q
70 NEXT x
```

e inserire un codice a ogni richiesta di input. Al termine, salvate su nastro con SAVE "nome" CODE 64256,800, quindi caricate il basic.

(Luca Bonaventura - Corsico MI) 1. Insisti con l'edicolante perchè richieda la rivista. 2. Non ci risulta che si trovi altro, anche perchè il software per lo ZX81 si va esaurendo sul mercato. 3. È stato pubblicato sul n. 10 (pag. 48).

(Giuseppe Cardella - Trapani) Come fai a dire che la tastiera non c'entra, se sono proprio i tasti che non funzionano? Sembra il tipico difetto da pista interrotta. Per i ricambi bisogna rivolgersi a chi ha venduto lo ZX81.

Posta Adventures

Vorrei alcuni consigli per l'avventure Valhallia della Legend.
(Mario Rambaldi - Napoli)
Mi dispiace deluderti, ma VALHALIA non è un vero e proprio avventure, non rientra quindi tra i programmi che io conosco. Pubblichiamo il tuo indirizzo per scambiare consigli e osservazioni con altri appassionati di questo programma: M. Rambaldi, via Acitilio 160, 80128 Napoli.

1) **The Hobbit: come fare ad uscire dalla caverna dei Goblin?**
2) The Hobbit: come far si che il drago e il bardo si incontrino?
3) Circus: quale è lo scopo del gioco?
(Francesco Ghirotto - Ravenna)
1) Per uscire dalla caverna dei Goblin, dobbiamo essere alla presenza di Gandalf o Thorin e quindi digitare:
SAY TO THORIN (o GANDALF) "OPEN WINDOW"
SAY TO THORIN "CARRY ME"
SAY TO THORIN "WEST".
2) Bisogna indicare al Bard la strada che deve prendere, per esempio SAY TO BARD "NORTH".
3) Bisogna far saltare in aria il circo "maledetto" e fuggire, con l'auto, prima dell'esplosione.

Come si entra nella "BOAT" in The Hobbit? *(vari lettori)*
Per entrare nella "BOAT" dobbiamo, dopo aver preso la corda e aver raggiunto la sponda del fiume, eseguire questi comandi:
THROW ROPE ACROSS
PULL ROPE
CLIMB INTO BOAT
CLIMB OUT.

Perchè rispondendo nel giusto modo alle domande di Gollum (ho letto il libro) in The Hobbit, vengo sempre strangolato?
(Alberto Braghetto - Rimini (Fo))
La cosa migliore è non perdere tempo a rispondere, ma uccidere direttamente Gollum.

1) In "THE HULK" come devo fare per liberarmi delle corde che mi tengono legato alla sedia?
2) In "ORACLE'S CAVE", non ho capito l'uso che si deve fare della chiave magica. *(Paolo Chirco - Firenze)*
1) Per trasformarsi nel mostro verde, e per poter quindi strappare la corda, bisogna causarsi del dolore mordendosi le lingua: BITE LEAP.
2) ORACLE'S CAVE non è un adventure.

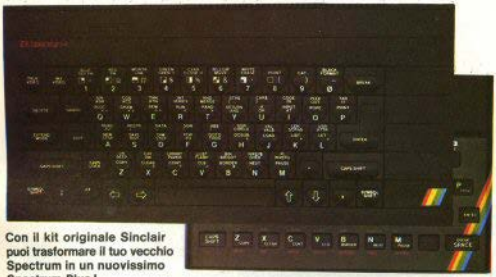
Per prima cosa, come fai a risolvere tutti gli adventure?
1) In "The eye of Bain", non riesco a liberarmi dalle catene
2) Come uscire dalle fognature e dal labirinto degli specchi in "Wax works"? *(Andrea Garau - Bolzano)*
Per quanto riguarda la soluzione degli adventure, devo dire che il lavoro non è tutto mio. Siamo un gruppo di amici che si riunisce spesso per risolvere questi enigmatici programmi. Anzi colgo l'occasione per ringraziare Fulvio Cerlesi, Benedetto Dell'Olmo e Francesco Baroni per la loro preziosa e indispensabile collaborazione. Ma passiamo alle risposte.
1) In Eye of Bain, devi sollevare il palo con il verbo LIFT.
2) Per quanto riguarda i labirinti, possono essere risolti con il sistema di "pollicino", già descritto sul numero 10 di SINCLAIR COMPUTER.
Per quanto riguarda il vocabolario, ti consiglio l'acquisto di uno più completo.

Vorrei porti alcune domande su "The golden baton":
1) quale è lo scopo?
52) come si apre la porta gigantesca?
53) come si va nel lago? *(Antonio di Cristoforo - Manfredonia FG)*
1) Lo scopo di questo adventure è quello di trovare la Bacchetta d'oro
2) per aprire la porta, bisogna pulire l'anello: POLISH RING. Apparirà una chiave. Digita quindi UNLOCK DOOR e poi GO DOOR.
3) Per superare il granchio gigante, e quindi raggiungere il lago, devi offrirgli il corpo della lumaca (GIVE SLUG) e quindi GO LAKE.

Ho subito delle difficoltà nell'iniziare "TIME MACHINE". Come superare la palude?
(Alberto Okely - Cernusco S/N MI)
Dal luogo di partenza, devi seguire queste precise direzioni: nord, ovest e quindi sud. Vedrai che da qui potrai continuare.

Scrivete a
Giuliano Boschi
Via F. Massi 12
00152 ROMA
Attenzione:
si risponderà alle lettere solo sulle pagine della rivista; non è possibile inviare risposte private.

Trasforma il tuo Spectrum in Spectrum +



con sole
87.500 LIRE + IVA

- Tastiera professionale di 58 tasti con 17 nuovi comandi
- Componenti di aggiornamento all'ultima versione dell'hardware: la Issue 6B
- Compatibile con tutto il software e le periferiche esistenti
- Completo di istruzioni, guida in italiano, cassetta dimostrativa

Con il kit originale Sinclair puoi trasformare il tuo vecchio Spectrum in un nuovissimo Spectrum Plus!
Il kit contiene oltre alla tastiera in robusto materiale antiurto, i chips che aggiorneranno all'ultima versione, la Issue 6B, il tuo Spectrum. Il nuovo software e le nuove periferiche saranno realizzati proprio in funzione della Issue 6B, solo così quindi puoi garantirti un computer al passo coi tempi! Potrai finalmente usare il tuo Spectrum con la facilità con cui si digita su una macchina da scrivere, e il nuovo tasto di reset ti consentirà di cancellare il programma in memoria senza scollegare l'alimentatore. Il kit contiene anche le istruzioni in italiano per il montaggio, facilissimo da eseguire: basta saper saldare qualche filo. Puoi farti aiutare da un amico o anche da un comune tecnico radiotelevisivo. Se hai poi ancora soltanto uno Spectrum 16 K, questa è l'occasione per avere l'espansione di **memoria a 48K** a un **prezzo favoloso:** solo **49.000 lire** + iva.

Ed ecco finalmente le esclusive confezioni di singole **cartridge** per **microdrive,** originali **Sinclair,** a un **prezzo** veramente **da sballo!!**

LIRE 5.900 + IVA

ZX SPECTRUM EXPANSION SYSTEM AL PREZZO DI 278.000 LIRE + IVA

La confezione contiene:

INTERFACCIA 1 indispensabile per il collegamento del Microdrive, munita inoltre di un'interfaccia RS 232.

MICRODRIVE, l'alternativa Sinclair ai floppy, che amplia le possibilità dello Spectrum con la ricerca velocissima dei dati sui particolari supporti magnetici (microcartridge).

SOFTWARE su 4 cartridge, con:
- Word Processor 'Transword 2'
- Masterfile file system
- Game designer
- Ant Attack

In omaggio il libro della Jackson sul Microdrive e l'Interfaccia 1

INTERFACCIA PROGRAMMABILE PER JOYSTICK

A SOLE
L. 59.750
+ IVA

La **Stonechip Electronics** è stata la prima ditta a realizzare per lo Spectrum un'interfaccia programmabile, considerata in Inghilterra fra le migliori. Facile da usare, si programma da tastiera. Garanzia 6 mesi.

50 FANTASTICI VIDEOGIOCHI 50

Registrati su nastro, della software-house inglese Cascade Games, all'**inverosimile prezzo** di sole **Lire 25.000 + IVA !!**

12 CASSETTE C-20 A LIRE 12.600 + IVA
Nastro professionale AGFA PE619, box trasparente in Kofil assemblato con 5 viti, scorrimento su perni in acciaio lubrificati.

Spedire il presente **MODULO D'ORDINE,** o fotocopia, in una busta chiusa, unendo L. 3.000 in francobolli per le spese postali.

Spett.le **APCO s.r.l. - Cas. Post. 239 - 10015 IVREA (TO)**
desidero ricevere quanto da me contrassegnato con X. Pagherò direttamente al postino gli importi qui elencati, che sono comprensivi di IVA e di spese di imballo e contrassegno. s

☐	Kit trasformazione in Spectrum Plus	a L. 105.000
☐	Espansione di memoria RAM a 48K	a L. 59.000
☐	ZX Spectrum Expansion System	a L. 330.000
☐	Raccolta di 50 VIDEOGIOCHI su nastro	a L. 30.000
☐	N..... confezioni di 12 Cassette C-20	a L. 16.000
☐	N..... confezioni singole microdrive cartridge	a L. 7.500
☐	Interfaccia programmabile per Joystick	a L. 72.000

Nome e Cognome _____

Via _____

Città _____

Robert L. Swarts
ZX80 e ZX81 come strumenti di controllo
Franco Muzzio & C. Editore, 1984 - p. 224 - L. 15.000

"Siete stanchi di giochi? State cercando qualcosa di veramente nuovo e diverso? L'avete trovato". Questo, in sintesi, il contenuto della simpatica introduzione, che dovrebbe suonare come un invito a nozze per tutti gli smanettoni che hanno ancora nel cassetto gli ZX80/81, sopo essere passati a computer superiori o per abbandono.

Occorre qualche conoscenza di elettronica, ma del tutto elementare; si prosume (ed è proprio il minimo) che non abbiate problemi con il basic Sinclair, che sappiate leggere semplici circuiti, e quanto al linguaggio macchina, può essere finalmente l'occasione per impararlo, ma non è indispensabile per la maggior parte dei progetti presentati.

Un notevole pregio del libro, scritto con toni divertenti e spesso ironici, com'è stile tipicamente americano, è di non essere inutilmente trionfalistico: fin dalle prime pagine si mettono in evidenza i limiti oggettivi della macchina, e i progetti presentati sono realistici sia per il costo che per il livello tecnico.

Molto valido l'insieme anche dal punto di vista didattico, accurata la traduzione.

M. Henrot - J. Boisgontier
Lo ZX Spectrum per tutti-iniziazione + programmi
EPSI/Editsi, 1985 - p. 130 - L. 13.000

Sempre piuttosto faticose, invece, le traduzioni in questa collana della Editsi, riproposta in italiano di una serie di manuali di informatica che in Francia va per la maggiore (almeno così ci riferiscono): i testi risentono sensibilmente dello sciovinismo linguistico del paese d'origine, in cui a un ormai universale "computer" si oppone ostinatamente "ordinateur".

Quanto al contenuto, il libro si propone come una delle tante alternative al manuale che accompagna lo Spectrum, inserendosi in classifica in posizione sicuramente dignitosa. L'impostazione è prevedibile (non c'è molto da inventare): primi contatti con lo Spectrum, la tastiera, lo schermo, struttura di un programma, i comandi del basic, gli UDG, la grafica, etc.

Neil & Pat Cryer
BASIC Programming on the QL
Prentice Hall International, 1985 - p. 264

Il fatto che per uno dei primi 'veri' manuali sul Superbasic si sia scomodata una casa editrice di prestigio come la Prentice/Hall Int. sta a dimostrare la serietà con cui in Gran Bretagna stanno amministrando il 'fenomeno QL'... a differenza di quanto avviene, per ora, da noi, dove le potenzialità della macchina vengono misconosciute, la pubblicità è poca, la dotazione di accessori, software e manuali procede con lentezza tutta burocratica.

Un vero manuale, dicevamo: non poche notizie raffazzonate, mas oltre 260 pagine di trattazione rigorosa, con moltissimi esempi puntigliosamente spiegati.

Ogni argomento affrontato è accompagnato da un paragrafo di esercizi e proposte di problemi da risolvere, dei quali si fornisce la soluzione (o una possibile soluzione) al termine di ciscun capitolo.

In molti dei quali si trova un paragrafo intitolato "Punti da ponderare", seguito da una "Discussione sui punti da ponderare": vi si affrontano problemi di programmazione dal punto di vista teorico (con esempi di soluzioni), analizzando anche i criteri che hanno ispirato gli estensori del Superbasic e del QDOS.

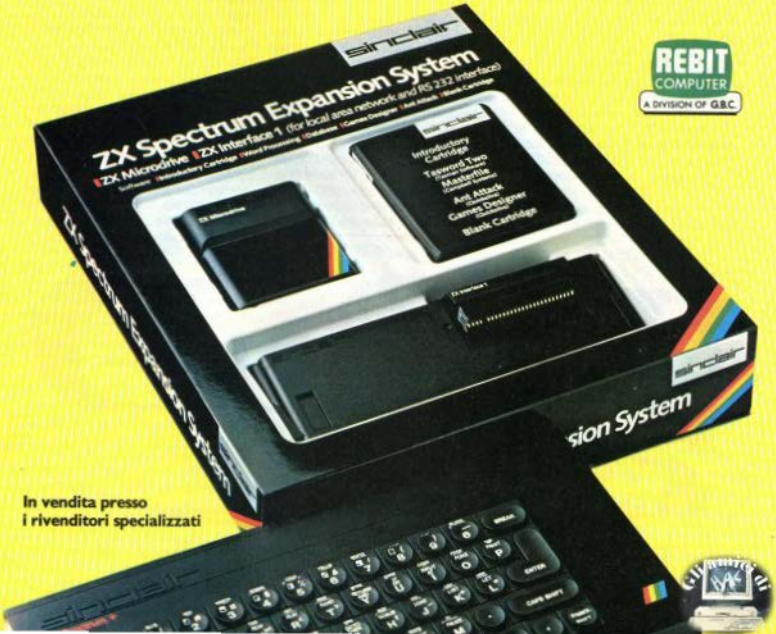
Un libro che non dovrebbe mancare all'utente del QL: auguriamoci che venga sollecitamente tradotto o almeno distribuito in edizione originale. Una nota di merito anche alla veste grafica e all'accuratezza dell'impaginazione.

ZX Spectrum Expansion System

L'alternativa della Sinclair ai floppy disc

Lo ZX Spectrum Expansion System contiene:

- **Uno ZX Microdrive** - Che amplia la possibilità dello ZX Spectrum in quei settori, come quelli della didattica e delle piccole applicazioni gestionali, dove è necessaria una veloce ricerca delle informazioni memorizzate su un supporto magnetico.
- **Una ZX Interface 1** - Indispensabile per il collegamento dello ZX Microdrive. Incorpora una interfaccia RS 232 e un sistema di collegamento in rete locale.
- **Quattro cartucce Microdrive comprendenti un programma di:**
  - Word processor «Tasword Secondo»
  - Masterfile filing system
  - Inventore di giochi
  - Le formiche giganti
- Un programma dimostrativo del Microdrive
- Documentazione per il collegamento, il funzionamento e altre descrizioni tecniche.
- Cavi di collegamento allo ZX NET che può collegare fino a 64 computer ZX Spectrum o QL.



REBIT COMPUTER
A DIVISION OF G.B.C.

In vendita presso i rivenditori specializzati

Arrow of death

di Giuliano Boschi

Dopo aver completato la prima parte di Arrow of death (S.C..n. 11), carichiamo la seconda puntata di questo adventure nella memoria del nostro Spectrum. Siamo pronti a continuare la storia.

Ci troviamo ai bordi di una palude (1). Digitiamo INVENTORY, e notiamo che abbiamo con noi alcuni degli oggetti trovati nella prima parte della storia: una punta di freccia, un ramo di salice, una piuma d'aquila (con le quali dovremo costruire una freccia magica e mortale, con cui uccidere il malvagio Xerdon) e una spada. Rechiamoci con circospezione a est (E). Una scialba pianura (2) si apre di fronte a noi, offrendoci la possibilità di scegliere tra numerose vie da percorrere. Puntiamo verso N. Qui (3) troviamo un misterioso cespuglio. Prendiamolo (GET SHRUB) e ci troviamo in mano soffici foglie.

Ritorniamo a S e poi ancora S. Siamo al luogo (4); chissà perchè, dobbiamo scavare (DIG); troviamo una utilissima pietra focaia, che ci affrettiamo ovviamente a prendere con noi (GET FLINTSTONE).

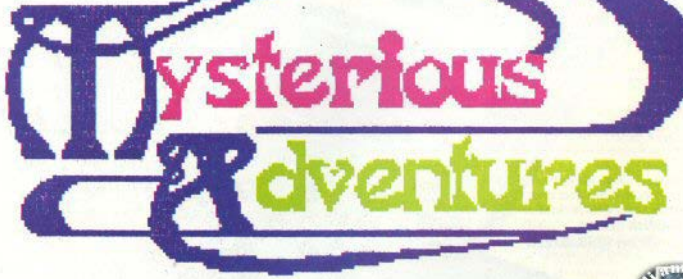
Torniamo al luogo 2 (N); spostiamoci a E, N, N: una sorgente sgorga dalle rocce (7) e ci blocca il passaggio. Non resta che tornare al punto di partenza (S, S, W, W,); andiamo poi a nord, per due volte (N, N). Questa volta è un burrone (9) che non ci permette di proseguire, ma niente paura, possiamo sempre provare a saltare (JUMP). Raggiunto un punto più basso, davanti a noi troviamo un ponte di corde.

Tentiamo di attraversarlo, sperando che possa reggere il peso (GO BRIDGE). Mentre stiamo in bilico sul ponte (11), un uccello vola sopra di noi e lascia cadere qualcosa. Torniamo indietro (S) scendiamo (D) ancora. Ecco la cosa: un elmetto di metallo. Raccogliamolo (GET HELMET). Notiamo anche la presenza di un crepaccio. Avviciniamoci (GO CREVICE), ed entriamo in una caverna (13). Troviamo una lampada ad olio. Aggiungiamola ai nostri averi (GET LAMP).

Torniamo sul ponte (N, U, GO BRIDGE), andiamo a N, raccogliamo (GET WEED) l'erba che si trova in questo prato (12). Saliamo (U, 13). Una inspiegabile griglia di metallo sembra impedire di nuovo ogni via di uscita (in questa seconda parte alcuni passaggi sono molto poco logici, e occorre molta fantasia).

Ma coraggio, non tutto è perduto. Rechiamoci di nuovo sul ponte (D, S). Il computer ci fa notare la presenza delle corde che sorreggono il ponte. Afferriamole saldamente (HOLD ROPE) e, con la spada, tagliamole (CUT ROPE). Il ponte si sfascia e precipitiamo rovinosamente, ma le foglie che abbiamo con noi permettono un atterraggio morbido (sic!), attenuando gli effetti della caduta.

Ci troviamo su di un tratto di roc-



cia (14). Lasciamo le ormai inutili foglie (DROP LEAVES) e raccogliamo il pezzo di corda, che ci ha seguito nella caduta (GET ROPE).

Andiamo verso l'arco di pietra (GO ARCHWAY) e da qui (15) a E.

Il buio ci avvolge, ma con lampada e pietra focaia non è un problema fare luce (LIGHT LAMP): discerniamo un lungo corridoio (16); continuiamo a E: una porta, saldamente chiusa (17), costringe a tornare indietro (W, S).

Proviamo a N (18) e ancora N. Incontriamo una scala (19) che sale e scende: saliamo (U); una macabra scena ci si presenta di fronte. Un guerriero morto e un enorme aquilone. Esaminiamo il guerriero (EXAMINE WARRIOR): ha il cranio sfondato e indossa un'uniforme. Lasciamo momentaneamente la corda (DROP ROPE), prendiamo la divisa e indossiamola (GET UNIFORM, WEAR UNIFORM).

Ora scendiamo due rampe di scale (D, D): giungiamo in una stanza di pietra (21), alla presenza di una guardia e di una gigantesca ruota di metallo. La guardia, vedendo la nostra uniforme, ci ignora. Proviamo a girare la ruota: è troppo pesante. Abbiamo una strana erba: mangiamo un po' (EAT WEED)... Ci sentiamo subito più forti.

Proviamo nuovamente a girare la ruota (TURN WHEEL), una sola volta o un numero dispari di volte. Sentiremo un rumore di meccanismo, ma non accade niente di visibile. Andiamo a E; anche qui (22) una pesante porta ci impedisce il cammino. Torniamo al luogo 21 (W, U, U), lasciamo l'erba e l'uniforme (DROP WEED, REMOVE UNIFORM, DROP UNIFORM), prendiamo l'aquilone e la corda (GET KITE, GET ROPE), quindi W. Ci troviamo su di una piattaforma (23), sotto di noi si apre un baratro.

A mali estremi, estremi rimedi: tenteremo con l'aquilone a mo' di deltaplano. Indossiamo il casco (WEAR HELMET) e saltiamo (JUMP). È fatta, atterriamo sani e salvi nel luogo 9. Lasciamo aquilone ed elmetto (DROP KITE, REMOVE HELMET, DROP HELMET), andiamo al luogo 7 (S, S, E, E, N, N). Non vi è più la sorgente, ma solo fango. Ora si comprende che cosa è successo girando la ruota del luogo 21: abbiamo interrotto la fuoriuscita dell'acqua.

Entriamo nel fango (GO MUD, 24); c'è una piastra nel muro. Cerchiamo nel fango (EXAMINE MUD) e troviamo una leva di metallo, tiriamola (PULL LEVER) e la piastra scorre, rivelando un'apertura nella roccia, che attraversiamo (GO OPENING). Ci troviamo ora all'interno di una costruzione (25), proseguiamo verso est (E, 26), quindi nord (N, 27). Incontriamo uno strano animale, che dobbiamo purtroppo uccidere (KILL ANIMAL): lascerà cadere una preziosa chiave, che ci affrettiamo a raccogliere (GET KEY).

Andiamo a S, E, S, quindi saliamo (U). Troviamo una porta sprangata, sblocchiamola (UNBOLD DOOR) entriamo (GO DOOR). Siamo di nuovo nel luogo 22. Raggiungiamo la stanza 17 (W, U, S, S, E, E): ora abbiamo la chiave, possiamo sbloccare anche questa porta (UNBLOCK DOOR) ed entrare (GO DOOR). Si tratta di un magazzino (30). Posiamo la chiave (DROP KEY). Non possiamo certo lasciare che pane e formaggio restino a lì ad ammuffire, prendiamoli (GET BREAD, GET CHEESE).

Torniamo al luogo 28 (N, W, S, N, N, D, E, GO DOOR, D, N), quindi a nord (N), dove c'è una stalla (31). Un mulo affamato ci guarda tristemente: diamogli un po' di cibo (FEED MULE). Ora ci guarda contento, viene vicino e ci segue affezionato (S, W, W, D); entriamo in un sotterraneo (32), vediamo una grata (EXAMINE GRATING): sotto di essa vi è un prigioniero. Leghiamo la corda prima alla grata e poi al mulo (TIE ROPE, TO GRATING, TIE ROPE, TO MULE).

Tiriamo vigorosamente il mulo (PULL MULE), la grata è divelta, entriamo nel buco (GO HOLE) (33). Troviamo Arnid, il costruttore di frecce. Osserviamolo più attentamente (EXAMINE ARNID), scopriamo che è privo di sensi. Dobbiamo andare a riprendere l'erba che abbiamo lasciato nella stanza 21 (U, U, E, E, S, U, GO DOOR, W, GET WEED) e torniamo alla 33 (E, GO DOOR, D, N, W, W, D, GO HOLE). Offriamo la potente erba a Arnid (GIVE WEED, TO ARNID), che si riprende e dice: «mi hai salvato, comandami».

Conoscendo ovviamente la sua maestria nel costruire le frecce, gli chiediamo di montarne una, con le parti di cui noi siamo in possesso (MAKE ARROW). Arnid fa la freccia e se ne va. Se digitiamo INVENTORY, vediamo che siamo in possesso della freccia magica (magical arrow).

Adesso dobbiamo trovare e distruggere Xerdon. Rechiamoci nel luogo 28A (U, U, E, E, S), quindi ancora a S. Ci troviamo in un tempio (34). Esaminiamo l'arazzo (EXAMINE TAPESTRY): c'è un bottone d'argento nel muro. Spingiamo (PUSH BUTTON) e, dal muro, esce uno strano altare, su cui (EXAMINE ALTAR) troviamo una candela. Accendiamola (LIGHT CANDLE) e, visto che siamo in un tempio, preghiamo (PRAY). Una colonna di fiamme si alza dalla candela; occorre ancora

un EXAMINE FLAME per scoprire che la fiamma sembra più debole. Rischiamo il tutto per tutto ed entriamo tra le fiamme (GO FLAME).

Istantaneamente ci troviamo su di un altare (35), in un posto che non è quello che abbiamo lasciato. Spengiamo la lampada (EXTINGUE LAMP), andiamo a S, S, E. Siamo all'esterno di una capanna (38), entriamo (GO HUT, 39). Troviamo pipa e tabacco, prendiamo anche questi (GET PIPE, GET TOBACCO), quindi N, W, S.

A un piccolo molo è attraccata una barca (40): montiamo (GO BOAT); su una barca (41) non possiamo non mancare i remi: GET OARS e remiamo (ROW BOAT). Ma, come lasciamo la riva, siamo inghiottiti da qualcosa e tutto diventa buio. Accendiamo la lampada (LIGHT LAMP): siamo nello stomaco di un enorme pesce (42) ...ricordate Pinocchio?

Intanto troviamo un logoro mantello, che subito raccogliamo (GET CLOACK) e indossiamo (WEAR CLOACK).

Carichiamo la pipa con il tabacco (FILL PIPE), accendiamola (LIGHT PIPE) e fumiamo per due volte (SMOKE PIPE, SMOKE PIPE). Il fumo riempie la pancia del pesce, che tossisce e ci espelle sbattendoci violentemente su una spiaggia sassosa (43). Lasciamo i remi (DROP OARS). Uno scheletro animato ci blocca il passaggio: per ora ignoriamolo, andiamo verso sud per due volte (S, S), quindi a W (46).

Raccogliamo la pietra che si trova a terra (GET ROCK), entriamo nella caverna (GO CAVE) (47). C'è una pala, prendiamola (GET SHOVEL) e scaviamo (DIG). Troviamo un candelotto di dinamite con un pezzo di miccia (GET DYNAMITE), continuiamo a N, notiamo un mucchio di sassi (EXAMINE CAIRN) e troviamo un sasso liscio e brillante.

Lasciamo la lampada, prendiamo lo strano oggetto (DROP LAMP, GET STONE), studiamolo (EXAMINE STONE): su di esso è rappresentata la figura di un mendicante (lo stesso della prima parte?) strofiniamo la pietra (RUB STONE): appare il mendicante, al quale diamo il sasso

(GIVE STONE). Lo strano personaggio prende la pietra, ci lascia un arco magico che ovviamente non lasciamo lì (GET BOW), e scompare.

Torniamo al luogo 43 (S, E, E, N, N); con la grossa pietra distruggiamo lo scheletro (SMASH SKELETON); il sentiero è ora percorribile (GO TRAIL), arriviamo al luogo 49; scaviamo di nuovo (DIG), entriamo nel buco che siamo riusciti a ottenere (GO HOLE). Da questo (50) andiamo a E, ma la via è chiusa (51).

È il momento di usare la dinamite: DROP DYNAMITE, accendiamo la miccia (LIGHT FUSE) e scappiamo subito a W. Aspettiamo il botto (WAIT), quindi torniamo a E. Un ampio varco ci permette di continuare questo nostro cammino che sembra senza fine (GO HOLE).

Siamo entrati in una costruzione (52). Continuiamo a N (53), quindi saliamo (U, 54): mentre una strana nebbiolina luccicante invade questo ambiente, continuiamo a N. Troviamo un organo a canne, diletto del padrone del palazzo (55); EXAMINE ORGAN, e troviamo uno spartito musicale.

Esperti musicisti, (Indiana Jones sa fare tutto, no?) possiamo suonare la musica dello spartito sull'organo (PLAY MUSIC)... il computer avverte che sta accadendo qualcosa. Infatti, tornando a sud (S), la nebbiolina è scomparsa e un corridoio ci permette di continuare il nostro cammino (GO CORRIDOR). Ancora due volte a nord (N, N) e avremo di fronte una vetrata (58); un ultimo passo a S.

Siamo in una stanza con una fessura nel muro (59). Esaminiamo la fessura (EXAMINE SLITS): dall'altra parte c'è una stanza (60) con Xerdon. Veniamo anche informati della possibilità di colpire da lì, il malvagio nemico. Non c'è allora tempo da perdere: incocchiamo la freccia, prendiamo la mira e lasciamo partire il mortale messaggio (SHOOT XERDON). La freccia magica non manca il bersaglio, Xerdon muore, il suo corpo scompare in una nuvola di fumo. È fatta: l'avventura è finalmente conclusa.

Ricordiamo ancora che chi ha domande e richieste di aiuto sugli adventure può scrivere direttamente a Giuliano Boschi, via F. Massi 12, 00152 ROMA.

Programmazione in assembly con lo Spectrum

di Gianluca Carri

Controllo programma

Appartengono a questa categoria le istruzioni che permettono di effettuare salti a *subroutine* o a determinate sezioni del programma principale, su condizione o in modo implicito, e quindi JP, JR, DJNZ, CALL e RST. La prima istruzione, JP *nn*, memorizza l'indirizzo *nn* nel registro PC, provocando il salto alla locazione di memoria *nn*. La versione JP *cc,nn*, (dove *cc* è uno dei codici di condizione Z, NZ, C, NC, PE, PO, M, P visti nel n. 8 di SC) permette di effettuare il salto solo se la condizione specificata è vera; per es. JP C, 8000H prosegue l'esecuzione del programma dalla locazione di memoria 8000H solo se il flag C è attivato, pvvero se l'ultima operazione di aritmetica effettuata ha generato un riporto.

L'istruzione JR è da un punto di vista logico equivalente a JP; la differenza consiste nel fatto che l'indirizzo della locazione di memoria a cui saltare non viene specificato direttamente, ma viene calcolato in modo relativo al contenuto del registro PC. Esempio: JR 40 salta in avanti di 40 bytes: JR -12 salta indietro di 12 bytes.

Vi sono diversi vantaggi che solitamente fanno preferire l'uso di JR. Anzitutto JR permette di scrivere programmi **rilocabili**, ovvero eseguibili correttamente a partire da qualsiasi indirizzo. Per ottenere tale risultato è necessario evitare qualsiasi riferimento di tipo assoluto a una particolare locazione: JP 500H indica in modo assoluto la locazione 500H, mentre JR 12 indica in modo relativo la locazione che segue di 12 bytes quella attualmente in esecuzione.

Altro considerevole vantaggio: JR è più corta di JP. Richiede infatti 2 bytes contro i 3 di JP.

Ogni medaglia ha però il suo rovescio e difatti JR affianca ai pregi alcuni difetti non trascurabili: può saltare al massimo 129 bytes in avanti e 126 indietro rispetto alla posizione corrente, viene eseguita più lentamente, e non consente di provare il flag P/V e il flag di segno.

Abbiamo infine 3 versioni speciali di JP: JP (HL), JP (IX), JP (IY) che saltano alla locazione specificata dal contenuto del registro fra parentesi.

L'istruzione DJNZ, già vista più volte, riassume gli effetti di DEC B e JR NZ, *d*.

Abbiamo poi CALL, anch'essa già incontrata, che alle funzioni svolte da JP aggiunge la memorizzazione nello stack del corrente indirizzo, affinchè un'istruzione RET possa far ritorno al punto di chiamata, come avviene in basic con le più familiari istruzioni GOSUB e RETURN.

L'istruzione RST è una forma abbreviata di CALL, avente il vantaggio di occupare un solo byte e di essere eseguita più velocemente. Vi sono però solo 8 possibilità: RST 0, RST 8, RST 10H, RST 18H, RST 20H, RST 28H, RST 30H, RST 38H, ciascuna delle quali corrisponde a una CALL all'indirizzo di memoria specificato. Normalmente la disponibilità di queste istruzioni è sfruttata dal costruttore del computer, che posiziona agli indirizzi di destinazione di RST i punti di ingresso di alcuni sottoprogrammi usati molto frequentemente: lo Spectrum non fa eccezione alla regola, quindi RST 1OH può essere usato per visualizzare sullo schermo il carattere contenuto nel registro A, RST 28H accede alle routines di calcolo in virgola mobile, ecc.

Controllo sistema

Esiste nello Z80 una ristretta fascia di istruzioni necessarie per un controllo sul funzionamento hardware della CPU e cioè NOP, HALT, DI, EI, IM 0, IM1, IM 2, IN, OUT.

NOP *(no operation)* non fa niente per un ciclo della CPU. Non è per questo inutile, serve sia a definire con precisione la temporizzazione di un programma, che a cancellare zone di memoria.

HALT sospende l'attività della CPU fino all'arrivo di un segnale di interrupt. Poichè nello Spectrum gli interrupt sono generati automaticamente 50 volte al secondo, HALT può servire a introdurre ritardi nel programma, oppure a sincronizzare un' uscita su video con la frequenza di quadro.

DI, EI rispettivamente disabilitano e abilitano la ricezione degli interrupt.

Per quanto concerne IM 0, IM 1 e IM 2 si veda SC n. 9 a pag. 7.

Per finire, IN e OUT servono per leggere o scrivere dati da/su un dispositivo esterno come una stampante, un joystick, la tastiera, ecc. Consideriamo le varie forme delle istruzioni disponibili.

Abbiamo IN A (N) che preleva nel registro A il byte inviato dal port di ingresso N, e IN *r*, (C) che preleva nel registro *r* il byte inviato dal port di ingresso C.

Per OUT abbiamo invece OUT (N), A e OUT (C), r che sono l'opposto delle precedenti in quanto inviano il byte verso il port d'uscita.

In particolare, OUT (254), A permette di cambiare temporaneamen-

te il colore del bordo (definito dai bit 0,1,2 del registro A) o di attivare/disattivare l'altoparlantino interno (bit 4).

IN è frequentemente usato per leggere la tastiera: si veda a proposito SC n. 10 a pag. 7.

Le versioni "speciali" IND, INDR, INI, INIR, OUTD, OUTDR, OUTI, OTIR sono usate solo per trasferire grandi masse di dati dalla memoria a una periferica (es. Microdrive) e viceversa, a velocità particolarmente elevata.

Strutture di Dati

Sicuramente l'elemento che determina la funzionalità e l'efficienza di un programma Assembly è il sistema con cui sono organizzati i dati che esso usa. Il problema è spesso sottovalutato dal neofita, in quanto lavorando in basic ci si abitua a fare uso di strutture di dati predefinite, o implicitamente connesse a certe istruzioni. Per esempio l'istruzione basic READ *n* garantisce al programmatore di prelevare un dato dalla lista specificata dall'istruzione DATA, e quindi di assegnare il dato alla variabile *n*.

Lavorando in Assembly non esistono strutture di dati predefinite, eccezion fatta per lo stack, per cui è il programmatore stesso che deve crearsele, ottimizzandole per ciascuna applicazione specifica. Per struttura di dati si intende un sistema attraverso il quale è possibile mantenere in modo ordinato un insieme di dati numerici o alfanumerici, e accedere ai singoli elementi che lo compongono.

La struttura di dati più semplice è la lista sequenziale; in essa i dati sono memorizzati uno dopo l'altro, in locazioni contigue di memoria. Lo *stack*, le cose *(queues)* sono spesso organizzati come liste sequenziali. Per accedere ai dati si utilizza solitamente un *puntatore*, ovvero un registro che punta all'inizio della lista, e che viene quindi incrementato fino a contenere l'indirizzo del dato voluto. Si tratta anche della struttura quasi sempre usata nell'ambito delle cosiddette *lookup tables*. Esempio:

```
25000 GROP Dato 0
25004 ABCD Dato 1
25008 ALEP Dato 2
25012 1234 Dato 3
```

La struttura di dati appena vista è organizzata come lista sequenziale a partire dall'indirizzo 25000. Usando la tecnica delle *lookup tables* (in cui la risposta a un problema può essere fornita semplicemente selezionando un dato all'interno della tabella, senza effettuare calcoli matematici) potremo localizzare il dato 1 (ABCD) con il seguente sottoprogramma (ammettendo di avere nel registro A il numero del dato):

```
LOOKUP  LD   L,A
        LD   H,0
             ;numero entrata
        ADD  HL,HL
             ;HL=HL*2
        ADD  HL,HL
             ;HL=HL*2
        LD   DE,25000
             ;inizio tabella
        ADD  HL,DE
             ;HL=indir.dato
        RET
```

Potrete verificare mentalmente che chiamando LOOKUP con A = 1 avremo in uscita HL = 25004, ovvero l'indirizzo del dato n. 2. Questa tecnica è usata molto spesso per la sua facilità di implementazione e per la sua efficienza in molte situazioni.

Se per esempio vogliamo scrivere un programma per generare suoni e ci necessitano le frequenze delle note fondamentali per ricavare tutte le altre, risulterà vantaggioso memorizzare tali valori in liste sequenziali piuttosto che ricalcolarli ogni volta.

Spesso le liste sequenziali sono ordinate, ovvero un dato è minore o maggiore del successivo secondo un ordine numerico, alfabetico o di altro tipo. Questo consente di localizzare con maggiore rapidità i dati: un dizionario è una lista sequenziale ordinata alfabeticamente, nella quale il dato voluto può essere trovato scartando tutti quelli che, nell'ambito dell'ordine (alfabetico) usato, risultano essere maggiori o minori.

Un'altra importante struttura di dati è la lista collegata, o *linked-list*. In questo caso i dati non sono memorizzati sequenzialmente, e possono differire in lunghezza. Il concetto fondamentale risulta nel fatto che il primo dato di una lista collegata è l'indirizzo dell'elemento successivo. Tale elemento potrà quindi risiedere fisicamente in una locazione non necessariamente contigua a quella dell'elemento precedente, ma risulterà, secondo l'ordine logico determinato dalla struttura, conseguente al dato che ne ha fornito l'indirizzo.

Il vantaggio fondamentale di una struttura simile è che non importa spostare dati in memoria: se vogliamo cancellare, inserire, spostare un dato non è necessario modificare fisicamente il dato stesso, ma solo l'indirizzo di collegamento.

Lo svantaggio risiede nella necessità di usare due bytes per l'indirizzo di *link* di ciascun dato nella lista: è ovvio che se abbiamo a che fare con dati lunghi solo un byte l'uso di una struttura *linked list* è assurdo.

Altre strutture frequentemente usate sono quelle ad albero, necessarie per stabilire delle relazioni di ordine gerarchico fra i dati: ciascun dato punta generalmente a più dati i quali puntano a loro volta ad altri dati. Si tratta di una procedura particolarmente utile quando si devono conciliare procedure particolarmente intricate con una relativa semplicità dell'algoritmo del programma.

Vorrei sottolineare che quasi sempre le strutture standard sono inefficienti nell'applicazione particolare di un programma in Assembly: il programmatore spesso deve usare nuove varianti di tali strutture, per giungere ad un buon programma: se fosse necessario effettuare un'animazione di figure sullo schermo dovremmo progettare una struttura di dati che consenta di localizzare rapidamente la posizione di ciascuna figura, di stabilire la direzione, la velocità e altri parametri simili. Se la nostra animazione prevedesse il movimento simultaneo di 50 elementi, giungeremmo quasi sicuramente a un risultato assai scadente se la nostra struttura fosse mal progettata.

Ritorna in edicola

VIDEO BASIC

Il corso più entusiasmante su cassetta del Gruppo Editoriale Jackson per **Commodore 64, VIC 20 e Spectrum**

200.000 copie vendute

del 1° fascicolo della prima edizione

Ogni lezione uno spettacolo

Con la 1ª lezione una cassetta giochi compresa nel prezzo

Il corso è composto da:
20 fascicoli + (Quattordicinali)
20 cassette +
5 splendidi raccoglitori

Oggi è davvero facile imparare il Basic. Con Video Basic il corso su cassetta che ti permette di programmare subito il tuo computer. È facile: tu chiedi, lui risponde, tu impari. Passo dopo passo. Sul tuo schermo appaiono le domande, le risposte, gli esercizi e tu, senza fatica, presto e bene, impari a conoscere e programmare il tuo computer, sia esso un VIC 20, un Commodore 64 o un Sinclair.
Video Basic è in edicola.
Provalo subito.
Ogni lezione è uno spettacolo.

Oggi il Basic si impara così.
Video Basic, il corso su cassetta per parlare subito col tuo computer.

Video Basic
per imparare non solo il Basic.

Un'altra grande idea firmata

GRUPPO EDITORIALE JACKSON

Milano•San Francisco•Londra•Madrid

Accesso casuale ai Microdrives-II

di D. Bolla & R. Zonin

Dopo aver visto insieme il modo per leggere dal microdrive in modo random, parliamo ora di come *scrivere random*. A chi fosse sfuggita la prima parte, raccomandiamo di andarla a leggere sul numero 12.

Scrittura ad accesso random

Quando abbiamo affrontato il problema della scrittura random per la prima volta, pensavamo che fosse indispensabile il ricorso al linguaggio macchina; ciò si è rivelato falso e siamo in grado di presentare una serie di routines tutte scritte in basic, e quindi facilmente comprensibili da chiunque abbia un minimo di conoscenza del sistema di i/0 dello Spectrum, e in particolare delle variabili di sistema dell'Interface 1. Un loro elenco commentato è stato pubblicato sempre sul numero 12.

Sapete già (e se non lo sapete ve lo diciamo noi) che lo Spectrum, per ogni canale aperto, crea una mappa, nella quale mette tutte le informazioni relative a quel canale. Per i canali collegati ai microdrives viene creata anche una mappa del nastro, che tiene traccia di quali siano i settori liberi e quali quelli occupati. Avrete capito a questo punto dove sta il trucco: per scrivere "random" in un settore abbiamo bisogno di:

a) poter manipolare la mappa del microdrive in modo tale da ingannare lo Spectrum quando va a scrivere i dati, per esempio facendo apparire libero un settore occupato;

b) sapere quali siano i *flag* (= indicatori; letteralmente bandiere) che mostrano se un file è aperto in scrittura o in lettura;

c) sapere quale è il numero di settore nel quale è memorizzato un blocco di un file;

Vediamo quindi com'è organizzata la mappa del microdrive e come bisogna manipolarla per ottenere ciò che vogliamo.

La mappa è composta da 32 bytes, che partono dall'indirizzo 23792 compreso in poi (per un solo canale aperto). Ogni singolo bit corrisponde a un settore sul nastro: il bit 1 del byte 1 corrisponde al primo settore, il bit 2 al secondo, il bit 1 del byte 2 al settore 9, e così via. Se un bit è settato a 1, il settore corrispondente è occupato; in caso contrario è libero. In totale quindi si possono individuare 32*8 = 256 settori.

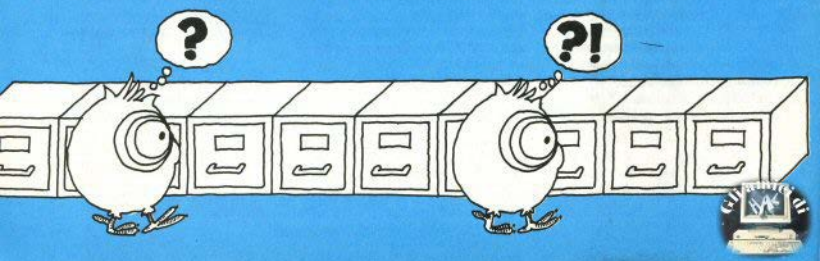
Modificare un file

Cominciamo ad esaminare il caso più semplice di scrittura random, e cioè la modifica di un qualsiasi settore di un file.

Dobbiamo far credere allo Spectrum che il settore che vogliamo modificare sia libero, e che sia l'unico settore nel quale può scrivere. Per modificare un blocco dobbiamo sapere in quale settore si trova (attenzione a non fare confusione fra il *numero d'ordine del blocco dati* e il *numero del settore* di nastro dove è registrato).

L'informazione che ci serve è contenuta nella locazione HDNUMB (byte 41 del canale), che indica proprio quale è il settore nel quale è registrato il blocco corrente. Una volta noto questo dato non è difficile creare una routine che liberi proprio il settore interessato. Questa routine, che si trova alle linee da 9000 a 9040, riceve in ingresso il numero del settore, e va a POKEare nel corrispondente byte della mappa il valore voluto.

A questo punto dobbiano solamente convertire il file da lettura a scrittura e viceversa. Niente di più facile: il flag che indica se un file è aperto in lettura o scrittura si trova nella variabile di sistema CHFLAG (byte 24 del canale). Per abilitare la scrittura basta eseguire una POKE 23844 + 24,255 e per la lettura una POKE 23844 + 24,254. Si deve tener



presente però che sia la lettura che la scrittura sono pilotate dalla variabile CHBYTE (11/12), che punta al prossimo dato da leggere o da scrivere.

Ciò vuol dire che se avete un file aperto in lettura e lo commutate in scrittura, scriverete nel file a partire dal dato puntato da CHBYTE. Questo fatto non provoca peraltro problemi nel caso che stiamo trattando (modifica di un file); infatti la modifica del file avviene in 3 fasi: lettura random del settore cercato, scrittura in RAM dei nuovi dati al posto di quelli vecchi, scrittura random del blocco dati modificato nel settore di nastro occupato dai vecchi dati.

Aggiungere dati a un file

Supponiamo ora che non ci basti sostituire vecchi dati con nuovi dati, e che il nostro obiettivo sia di aggiungere dati ad un file. Il metodo da seguire è praticamente lo stesso della riscrittura, solo che c'è un problema nuovo da affrontare: dove andiamo a scrivere? Quando abbiamo modificato la mappa del microdrive, prima abbiamo settato tutti i settori da 1 e poi abbiamo liberato quello che ci interessava; così facendo però abbiamo perso di vista quali erano i settori veramente liberi e quali quelli occupati. Per evitare questo inconveniente, prima di modificare la mappa originale dobbiamo creare una mappa alternativa, che permetta di ricordare i settori liberi e quelli occupati (dal file o da programmi).

Di questo si occupa la routine che si trova dalla linea 8000 alla 8040. La mappa alternativa viene messa in una matrice *p*. Tra parentesi, ricordiamo che i settori del microdrive sono numerati dallo 0 in poi, mentre le matrici sullo Spectrum hanno come indice inferiore 1; questo spiega l +1 o -1 presenti nelle routines del programma. La convenzione che si usa in questo programma è che il primo settore è il numero 1: tenetelo presente quando andate a smanettare per vostro conto.

Tornando al nostro file, poniamo di volergli aggiungere un blocco di dati. Per prima cosa carichiamo in memoria quello che è attualmente l'ultimo blocco del file. Modifichiamo la variabile RECFLG (byte 67) che, dopo ripetuti esperimenti, ci risulta essere semplicemente un Identificatore di fine file, ponendola a 0. In questo modo il blocco che era di fine file diviene per il sistema uguale agli altri blocchi (il valore 2 in RECFLG indica che questo blocco è l'ultimo del file). Infine riscriviamo il blocco.

Vediamo in dettaglio le operazioni da compiere per riscrivere un settore:

a) modificare la mappa

b) trasformare da lettura a scrittura

c) segnalare che il blocco è pieno con POKE 23844 + 13,3

d) digitare PRINT#4;" "

A questo punto il drive si metterà in moto e scriverà il blocco dati nel settore voluto. Ora in memoria avete un blocco vuoto; scrivete quello che volete nel modo standard (cioè con PRINT#4; "...."); quando avete finito POKEate in RECFLG il valore 2, scegliete nella mappa speciale un settore libero e ripetete la procedura di salvataggio.

Spiegazione delle routines

Come avrete notato il programma presentato è in buona parte strutturato a subroutines. Questo è stato fatto per cercare di renderlo facilmente leggibile e modificabile.

1) Copiatura della mappa dei microdrives: svolge il lavoro a cui abbiamo già accennato di copiare la mappa dei microdrives e di riprodurla in una matrice di 256 elementi. Il procedimento non è particolarmente complicato; può essere interessante la parte che fa la conversione del numero da decimale a binario: guardatevela.

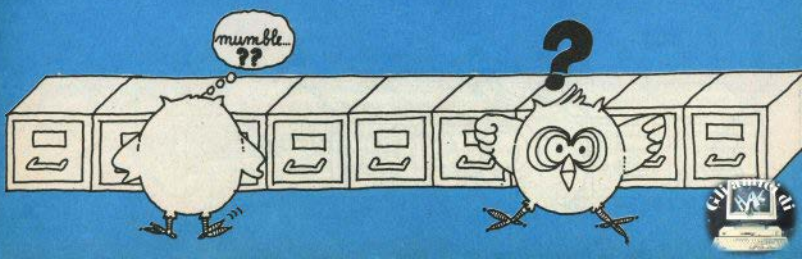
2) Trasformazione da lettura a scrittura. Anche qui niente di complicato: si tratta di sapere quali sono la locazione giusta e il valore giusto da scrivere con POKE. Notate che per ogni routine viene definita una variabile che punta, a seconda dei casi, alla mappa del microdrive oppure al punto di inizio delle informazioni relative al file in questione (è possibile infatti avere più file aperti contemporaneamente con mappe in diverse locazioni di memoria).

3) Trasformazione da scrittura a lettura. È tutto come la routine precedente, cambia solo il valore della POKE.

4) Lettura del numero del settore nel quale è registrato il blocco dati che abbiamo in memoria. Quell'uno che va aggiunto alla variabile (s) serve solo per fare in modo che ci sia una sola convenzione di numerazione dei settori. La variabile (s) è quella che va passata alla routine che libera il settore interessato.

5) Inserimento di EOF (End Of File). Serve solo nel caso scriviate un settore che è l'ultimo del file, e non è obbligatorio metterla. Però se non lo fate, sappiate che lo Spectrum vede quel file come NON chiuso, con perdita di efficienza in certe funzioni (per esempio DELETE).

6) Cancellazione di EOF. Questa viene richiamata regolarmente nella lettura random; quando leggete dei dati nell'ultimo settore, e arriva-



te alla fine dello stesso, se volete leggere un settore precedente il computer vi scriverà che il file è finito anche se tentate di modificare le variabili per proseguire. Lo Spectrum si comporta così vedendo in RECFLG un 2: basta trasformare questo 2 in uno 0 e tutto riprende a funzionare regolarmente.

7) Scrittura nel file del blocco corrente nel settore giusto. A preparare il settore giusto ci pensano le altre routines. Questa non fa altro che far credere al computer che il buffer è pieno; di conseguenza gli fa scrivere sul file tutto ciò che c'è nel buffer.

8) Lettura di un settore. Anche questa è una routine già vista in precedenza, e anch'essa deve "ingannare" il computer, facendogli credere che ha già finito di leggere il blocco corrente e che deve passare al successivo.

9) Liberazione di settore. Questa è totalmente nuova e serve a mettere a 0 (libero) il bit che indica il settore puntato dalla variabile (s). Non è complicata da capire, basta tenere presente come è organizzata la mappa del microdrive e ricordarsi che un settore è occupato se il corrispondente bit è 1.

10) Scrive 1 nella mappa del microdrive. Questa è forse la routine più banale: mette a uno tutti i bit della mappa del microdrive. Va sempre utilizzata prima della routine 9, come misura cautelativa.

11) Ricerca. Implementa la parte finale della lettura random (vedi Sinclair n. 12). Chiede in ingresso il numero del blocco e la posizione del dato, dopodichè va a leggere il dato stesso.

12) Setta a 512 il numero di caratteri inseriti. Lo scopo è di regolarizzare il puntatore del buffer, che in certi casi perde il controllo a causa dei nostri maltrattamenti...

13) Controlla la lunghezza della stringa da inserire nel buffer. Se la stringa fosse così lunga da superarne i limiti, verrà scartata e comparirà il messaggio "Illegal quantity ERROR". Questa routine quindi serve ad evitare di scrivere dati *a cavallo* di due settori del nastro. Ciò si è reso necessario soprattutto perchè le routines lavorano su un solo settore per volta. Non si tratta però di una limitazione grave: nell'uso normale i record logici da scrivere saranno difficilmente più lunghi di 512 bytes; inoltre, tenete presente che lo sfruttamento migliore delle memorie a settori si ha impiegando record logici la cui dimensione è un sottomultiplo della dimensione del blocco fisico.

14) Scrive nel buffer. Prima abbiamo detto che si può scrivere usando semplicemente PRINT#4, con cui però a volte si hanno dei problemi, perchè aggiorna i contatori dei dati inseriti. Abbiamo perciò incluso questa routine, che esegue una POKE della stringa D$ direttamente nel buffer. L'input tramite questa routine è quello da preferire.

15) Scrive sul nastro. È solo una sequenza di GOSUB, e serve a semplificare la vita a chi scriverà i suoi programmi includendo l'accesso random: infatti svolge automaticamente tutte le operazioni necessarie a scrivere fisicamente il buffer sul nastro del microdrive.

Ora avete tutte le informazioni necessarie per farvi il vostro *database*, utilizzando i microdrive quasi come dei dischi. In pratica, non vi sono limitazioni su come deve essere costituito un database, nel senso che per i dati non siete più vincolati alla memoria dello Spectrum.

Dopo aver lavorato per tutto questo tempo sui microdrive, pensiamo che il loro più grosso difetto non sia nella realizzazione o nel funzionamento ma nel prezzo delle cartucce, ancora altino, anche se più ragionevole di quello spropositato dei primi arrivi.

Qualche esempio d'uso

I tre brevi programmi che accodiamo alle routines sono un primo esempio di come si possono utilizzare tutte le primitive illustrate poco sopra. Tutti e tre vanno fatti partire dopo aver resettato il computer, per esser sicuri di non aver dimenticato nella memoria qualcosa che potrebbe disturbare.

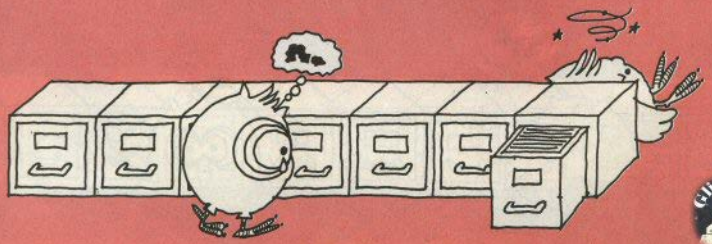
Programma Uno

Il primo programma, il più semplice, è un esempio dimostrativo e vi guida totalmente. Per cominciare crea un file composto di caratteri "#", e ve ne mostra i primi 100. Poi chiede di scrivere qualcosa; accontentatelo scrivendo un po' di caratteri, poi premete ENTER. Se quello che avete introdotto vi basta come prova, premete di nuovo ENTER. Se no inserite degli altri caratteri e premete ENTER.

Il computer scriverà nel file, a partire dal carattere 101, la stringa che avete inserito; poi passerà in lettura e vi mostrerà tutto il file con le modifiche da voi fatte (cercate di non esagerare con il numero dei caratteri per non incorrere in inconvenienti).

Programma Due

Il secondo programma è fondamentalmente simile al primo, ma siete voi che scegliete dove scrivere. Anche qui le solite raccomandazioni: non cercate di scrivere oltre l'ultimo settore generato dal programma (il terzo, in questo esempio), e se vedete apparire la scritta



illegal quantity ERROR non preoccupatevi, dovete solo introdurre una stringa più corta.

Programma Tre

Il terzo è più complesso e versatile. Prevede anche la possibilità di aggiungere un nuovo settore al file; questa funzione è di solito chiamata APPEND, ed è un po' complessa: per comprenderne il funzionamento esaminate il listato avendo presenti le spiegazioni che abbiamo dato alle varie routines. Ecco in breve come funziona la parte dalla linea 20 alla linea 70:

20 apre il file in lettura; analizza la mappa del microdrive; preleva le informazioni necessarie.

25 controlla se il blocco corrente è un blocco di fine file (ricordate la descrizione delle variabili di sistema)

26 "spazzola" tutti i blocchi del file e rimanda il controllo alla 25

28 setta il numero di bytes del blocco corrente

29 cancella l'EOF

30 scrive questo blocco così modificato sul nastro

40 resetta il numero di dati del blocco

50 inserisce l'EOF

55 inizializza una variabile fittizia

60 controlla nella matrice che contiene le informazioni relative alla mappa quale è il primo settore libero, e lo assegna alla variabile di canale adeguata.

70 continua fino a quando viene soddisfatta la condizione.

Dopo che è stato trovato il settore libero si passa alla routine di scrittura per inserire i nuovi dati nel nuovo blocco.

Conclusione

Da notare che questi files, creati in modo random, sono comunque leggibili serialmente con PRINT INKEY$#4; questo perchè il nostro sistema di gestione random non altera la struttura fisica dei files, ma solo il modo di accedervi, ingannando il sistema operativo.

Arriverci quindi alla terza puntata, in cui descriveremo un esempio pratico d'uso di queste routines: un *database* che sfrutterà appieno tutte le possibilità dei microdrives.

Soft_ware

Polinomi

di G. Aicini e S. Saisano

Breve programma per trovare velocemente le radici razionali di polinomi con grado massimo 100. La soluzione di uno dei più tediosi problemi di matematica avviene inizialmente per tentativi, sostituendo all'incognita i quozienti ottenuti dalla divisione tra i divisori del termine noto e quelli del coefficiente del termine di grado maggiore: una volta trovata la radice, il polinomio viene scomposto usando il metodo di Ruffini e viene cercata una nuova radice sul polinomio, di grado inferiore a quella originale, risultante dalla scomposizione; si procede in questo modo finchè non è più possibile trovare radici nel sottopolinomio attuale.

I coefficienti dei termini del polinomio devono essere numeri interi, e bisogna indicare con zero il coefficiente di eventuali termini mancanti nel polinomio. (list a pag. 29)

Grafici polari

di Stefano Pizzeghello

Normalmente le funzioni sono rappresentate nella forma *cartesiana*, secondo la quale ogni punto del piano è rappresentato da una coppia di numeri, che indicano la distanza in orizzontale (x) e quella in verticale (y) dall'origine.

Un punto nel piano può tuttavia essere rappresentato anche da una diversa coppia di numeri (r,t), dove *r* rappresenta la distanza assoluta del punto dall'origine, e *t* l'angolo tra la retta congiungente il punto e l'asse *x*: questa rappresentazione è detta «in forma polare», e ha relazione con la forma cartesiana per mezzo delle formule x = r*COS t e y = r*SIN t.

Come nella forma cartesiana possiamo definire una variabile in funzione dell'altra per mezzo di equazioni della forma y = f(x) (oppure con l'inversa x = f(y)), anche nella forma polare possiamo definire funzioni del tipo r = f(t) (molto meno usata è la forma t = f(r)).

Per mezzo di funzioni espresse in forma polare si riesce, tra l'altro, a tracciare grafici di funzioni non esprimibili in forma cartesiana, come la spirale o il cardioide.

Il programma consente proprio il tracciamento di funzioni espresse in forma polare, previa indicazione, oltre alla funzione, dei limiti inferiori e superiori dell'angolo t, il passo d'incremento dell'angolo stesso, e le dimensioni del grafico.

Un completo set di comandi consente di cambiare i parametri dell'elaborazione, nonchè di ingrandire o rimpicciolire il grafico, mantenendo la densità dei punti, o di stampare lo stesso grafico con tutti i parametri della funzione.

(list a pag. 29)

Integrali doppi

di P. Luigi Paccagnella

Gli studenti universitari delle facoltà scientifiche si trovano davanti, presto o tardi, gli *integrali doppi*, per mezzo dei quali viene calcolato il volume compreso tra il grafico di una funzione a due variabili (quindi a sviluppo tridimensionale) e il piano di base (dove la z, cioè l'altezza della funzione, è uguale a zero).

Soft_ware

Come chi mastica un po' di analisi matematica potrà capire, gli integrali doppi sono il corrispondente tridimensionale degli integrali semplici, quelli di una funzione con una sola variabile (nella forma y = f(x)), che essendo a sviluppo bidimensionale (su un piano x/y), calcolano l'area compresa tra la curva e l'asse delle ascisse.

Nel caso bidimensionale, inoltre, si integra la funzione su un segmento, mentre nel caso tridimensionale (cioè con funzioni a due variabili) la regione su cui integrare è una parte del piano.

L'area da integrare deve essere *normale* (perpendicolare) a uno degli assi del piano (*x* o *y*) e deve essere delimitata, nel caso sia normale all'asse x, a destra e a sinistra da due rette, sopra e sotto da due funzioni della variabile "x" (F superiore e F inferiore): il discorso è naturalmente simmetrico, potendosi scambiare la *x* con la *y* nel caso l'area su cui integrare sia normale all'asse delle *y*.

Un integrale doppio si può calcolare come integrale, rispetto a una delle due variabili, dell'integrale calcolato rispetto all'altra variabile: quindi il calcolo avviene in due fasi successive, in ognuna delle quali si calcola un integrale semplice (la prima volta quello della funzione da integrare, la seconda quello dell'integrale della prima fase); il problema è di essere il più precisi possibile nel calcolo degli integrali semplici, naturalmente facendo gli opportuni scambi di parametri tra un'integrazione e l'altra.

Il programma presentato risolve gli integrali doppi in circa 35-40 secondi, sia in forma cartesiana (usando nella formula *x* e *y* con il significato solito), che in forma polare (nella formula si dovranno ancora usare *x* e *y*, ma il loro significato è ora diverso: *x* al posto della *theta*, *x* al posto del *ro*), con una precisione oscillante tra l'ottava e la decima cifra decimale, comunque molto buona.

Il metodo usato appartiene ai metodi gaussiani: un integrale semplice di una funzione viene calcolato dividendo l'intervallo di integrazione in un numero di parti a piacere (dipende dalla precisione richiesta), moltiplicando il valore della funzione nei punti di divisione per opportune funzioni di opportuni valori o *pesi* (dati da certi tipi di polinomi, come quelli di Legendre) e sommando tutti i valori che così si ottengono. Nel programma presentato, l'autore divide l'intervallo in 12 parti e fornisce i *pesi* adatti al calcolo nel DATA di linea 20: attenzione a copiarla esattamente; i *pesi* sono poi inseriti negli array *w()* e *k()*.

L'integrale da A a B di una funzione f(x) è dato perciò da:

```
0.5*(B-A)*somma        (w(i)*f
(0.5*(B-A)*k(i) + 0.5*(B+A)))
```

(per *i* compreso fra 1 e 12).

Se uno dei limiti è indefinito, l'integrale tra il limite A e infinito è:

```
2 * somma ( ( w(i)/( (1+k(i)) ↑ 2) ) *
f( ( 2/(1+k(i)) + A + 1) )
```

(per *i* compreso tra 1 e 12)

Nel programma, alla richiesta dei limiti d'integrazione, si possono inserire limiti infiniti semplicemente inserendo *+i* o *-i*, a seconda che l'infinito sia negativo o positivo.

Il calcolo avviene quindi a *fette*, poichè il valore dell'integrale nei 12 punti di divisione sull'asse delle *x* non è altro che l'integrale della funzione, rispetto alla *y*, compreso tra la funzione superiore e inferiore e con la *x* corrispondente al punto di divisione (le *x* e le *y* vanno invertite se il discorso riguarda regioni d'integrazione normali all'asse delle *y*): in ogni caso il metodo è spiegato in ogni testo di Analisi Matematica (e chi è arrivato sin qui senza problemi ne avrà senz'altro letto uno...)

L'uso del programma è immediato: abbondare sempre con le parentesi nelle funzioni se non si è sicuri.

(list a pag. 31)

LIST control

di Giovanni Corbelli

Sulla scia dell'articolo di G. Carri apparso sul n. 7, ecco un altro programma che, sfruttando le caratteristiche dell'interfaccia 1, aggiunge utili comandi al basic dello Spectrum.

È possibile listare un programma (su schermo o su stampante) solo tra le linee specificate, con ovvia comodità, e in più si può fare il COPY del video a 24 righe (naturalmente questo comando dovrà essere dato da programma per essere utile, perchè se dato in modo diretto troverà le ultime due linee dello schermo bianche).

Viene anche fornito un secondo programma, sempre in codice macchina, che dà le stesse possibilità anche a chi non possiede l'Interface 1. Il modo di operare è solo accennato sul manuale dell'interfaccia: se un comando non è accettato dal basic Spectrum, con l'interfaccia 1 collegata si controlla se l'operazione richiede uno dei nuovi comandi disponibili, ed eventualmente lo si esegue.

Se invece è proprio un errore, cioè un'istruzione non prevista nè dalla sintassi dello Spectrum nè da quella dell'interfaccia 1, si salta alla routine il cui indirizzo è contenuto nella variabile di sistema VECTOR (23735/6; v. SC n. 12 pag. 45).

Se in questa variabile si mette l'indirizzo di una routine che abbiamo scritto noi e che controlla la sintassi dei nuovi comandi che abbiamo definito, possiamo estendere a piacere il basic dello Spectrum con qualsiasi istruzione ci venga in mente.

I comandi presentati qui sono:

COPY## (dopo il COPY bisogna battere un "cancelletto", cioè il CHR$ 35);

Soft_ware

Bubble-sort in assembly

di Luigi Callegari

Si copiano su stampante tutte le 24 linee dello schermo.

LIST (o LLIST) seguiti da un asterisco, dalla linea da cui deve partire il listato, e opzionalmente da TO (SYMBOL SHIFT + F) e dalla linea a cui terminare il listato; non specificando la linea di partenza verrà presa come default la linea 0; esempio:

LIST *30 TO 70 lista le linee da 30 a 70;

LLIST *30 TO 70 lo stesso, ma su stampante;

LIST *30 lista solo la linea 30, cosa molto utile in fase di correzione dei programmi perchè consente di editare subito la linea errata e di continuare poi con l'esecuzione.

LIST *TO 30 lista dalla linea 0 alla linea 30

È ancora possibile specificare il canale a cui deve andare il listato: OPEN #4, "n", 0: LIST#4, TO 30; CLOSE#4.

Per chi non ha l'interfaccia 1, gli stessi risultati sono ottenibili inserendo nel programma queste 2 linee (la RAMTOP sarà stata abbassata con CLEAR (st-1), dove *st* è l'indirizzo di caricamento del codice macchina):

```
10 DEF FN c() = USR (st + 68)
20 DEF FN l (x,y,z,) = USR st
```

RAND USR FN l (s,a,b) invierà il listato compreso tra le linee *a* e *b* al canale *s* (ricordiamo che il canale associato al video è il 2, alla stampante il 3).

RAND USR c() esegue invece il COPY a 24 linee.

Il programma caricatore 1 è per i possessori di interfaccia 1, il caricatore 2 per gli altri.

Il funzionamento della routine per l'interfaccia 1 è controllato dalla routine EXAM, il cui indirizzo di partenza è posto nella variabile di sistema VECTOR dalla routine INIT (chiamata con RAND USR *st* nel programma 1 alla linea 180); EXAM controlla la sintassi dei comandi e li esegue; per quanto riguarda il COPY a 24 colonne, viene dapprima eseguito il COPY normale chiamando la routine di ROM, poi si passa al COPY delle ultime 2 linee.

(list a pag. 32)

Dopo aver visto, nei precedenti numeri di Sinclair Computer, come sia possibile utilizzare differenti algoritmi per effettuare l'ordinamento di una serie di elementi, vediamo questa volta un'implementazione in codice macchina, estremamente più flessibile e veloce, pur utilizzando il metodo che in basic si rivela più lento, cioè il bubble-sort.

Il programma basic provvede al caricamento del codice macchina, che può essere salvato su nastro (o drive), pronto per essere inserito in altri programmi con un "LOAD" CODE. La routine è completamente rilocabile, cioè può essere posta ovunque in memoria ed essere quindi usata anche con 16K. Inoltre è protetta da qualunque tipo di errore ed è praticamente impossibile che mandi in "tilt" il calcolatore.

Nessun parametro deve essere passato al linguaggio macchina con POKE: il programma provvede a rintracciare in RAM la variabile alfanumerica di nome a$ e ricavare i dati relativi al numero di elementi da riordinare e alla loro lunghezza. I limiti posti a questi due numeri sono solo dovuti alla capacità di memoria dello Spectrum, teoricamente qualunque valore da 1 a 65535 è accettabile.

Si noti che la matrice a$ deve sempre avere due dimensioni, come di regola: la prima indica il numero di elementi, la seconda la loro lunghezza. Nel programma dimostrativo presentato la matrice è predisposta per accogliere 1000 nomi, lunghezza massima 29 caratteri ciascuno; alle linee 1420/1440 viene alterato il primo valore di mille con due POKE nella zona variabili che contiene quel parametro, affinchè risulti come se fosse stato impartito un comando DIM a$ (j,29), dove 'j' indica il numero di elementi effettivamente presente. Il '29' (la lunghezza) è lasciato immutato.

Quando la routine assembly è stata caricata (si consiglia, come al solito, sopra la RAMTOT), basta eseguire RANDOMIZE USR xxxxx, dove 'xxxxx' è l'indirizzo di caricamento. Il microprocessore Z80 si metterà al lavoro, esaminando circa 1400 caratteri al secondo, e fornendo sullo schermo il numero di nomi ancora da riordinare e una curiosa segnalazione colorata, dovuta al fatto che come memoria di lavoro viene utilizzata la 'mappa degli attributi colore'.

La routine assembly può anche riordinare in senso decrescente: è sufficiente effettuare POKE x + 163,48 (essendo 'x' l'indirizzo di inizio del caricamento); per riportarla al riordino crescente, dare POKE x + 163,56.

Si può inoltre eliminare la segnalazione numerica, aumentando ulteriormente la velocità di lavoro del programma:

POKE x + 155,0
POKE x + 156,0
POKE x + 157,0

Se si tenta di lanciare il programma con RANDOMIZE USR x e qualcosa non è stato preparato come richiesto (per esempio: non esiste la matrice a$, o non ha due dimensioni), si avrà un appropriato messaggio di errore e il ritorno istantaneo al normale editor del basic.

(list a pag. 34)

BOXE

di Andrea Riccluti

Soft_ware

Un divertente giochino interamente in basic, adatto anche ai possessori della versione 16K, su quella che qualcuno chiama ancora *noble art* (noi la confineremmo volentieri nei videogames).

L'autore è un ragazzo di soli 14 anni. Tutto quello che bisogna fare è sopravvivere 3 round contro il pugile-Spectrum (che naturalmente è nero...), spostandosi a destra e a sinistra sul ring con i tasti «N» e «M», in su e in giù con i tasti «Q» e «A», e colpendo l'avversario col tasto «W». All'inizio si selezionano la velocità di esecuzione e «la prontezza di riflessi» dello Spectrum (indici da 1 a 3, dove 1 sta per veloce, 2 per medio e 3 per lento), poi si combatte per i 3 round e alla fine chi ha colpito più volte l'avversario vince, ma se un boxeur è colpito più di 75 volte viene «contato» con un fischio e può finire K.O.

Tra un round e l'altro ci vengono spiritosamente mostrate le condizioni fisiche di ogni combattente, di cui viene visualizzato il viso più o meno incerottato: ricordate che attaccare sempre può far male, quindi cercate di non prendere tutti i colpi...

STUDIO D
PER NON SMARRIRE MAI IL FILO DEL DISCORSO.
STUDIO D
EMITTENTI RADIOTELEVISIVE INDIPENDENTI CHE SI FANNO SENTIRE.

studio d

CONCESSIONARI MEZZI RADIOTELEVISIVI

STUDIO D
Via Rossini 5 - 20122 MILANO
Tel. (02) 799.592-782.503

Microdrives random

```
7997 STOP
7998 REM  ROUTINES DI SCRITTURA
         RANDOM SU MICRODRIVES
7999 REM (C) 1985  ZONIN & BOLLA
       PER  SYSTEMS EDITORIALE
8000 REM 1=occupato 0=libero
8002 DIM p(256)
8003 LET m=23792
8005 FOR x=0 TO  1
8006 LET d=PEEK (x+m)
8010 FOR c=1 TO 8
8015 LET p(c+(x*8))=d/2<>INT (d/
2)
8020 LET d=INT (d/2)
8025 NEXT c
8030 NEXT x
8040 RETURN
8048
8049
8050 REM trasf. a scrittura
8060 LET can=23844
8070 POKE 24+can,255: RETURN
8077
8078
8079 REM trasf. a lettura
8080 LET can=23844
8090 POKE 24+can,254: RETURN
8098
8099
8100 REM lettore del record
8110 LET can=23844
8120 LET s=1+PEEK (can+41)
8122 REM  +1 e' per la matrice
8130 RETURN
8140
8141
8150 REM ins. EOF
8160 LET can=23844
8165 POKE can+67,2: RETURN
8168
8169
8170 REM cancella EOF
8175 LET can=23844
8180 POKE can+67,0:- RETURN
8190
8191
8200 REM go to print in file
8205 LET can=23844
8210 POKE can+12,1: POKE can+11,
255: PRINT #4;" ";: RETURN
8220
8221
8250 REM go to read in file
8255 LET can=23844
8260 POKE can+12,3: LET d$=INKEY
$#4: RETURN
8290
8291
9000 REM liberatore di settori
9001 REM richiesta var(s)
9005 LET m=23792
9006 LET sm=s-1
9010 LET s1=m+(INT (sm/8))
9020 LET s2=255-2^(sm-8*INT (sm/
8))
9030 POKE s1,s2: RETURN
9048
9049
9050 REM generatrice di 1
9060 LET m=23792
9070 FOR c=m TO m+31
9080 POKE c,255
9090 NEXT c
9095 RETURN
9098
9099
9100 REM cercatore del dato
9101 REM richiesto N.blocco (b)
9102 REM  e N. del dato   (d)
9105 LET can=23844
9110 POKE can+13,b-1
9115 GO SUB 8250
9120 POKE can+11,(d-256*(INT (d/
256)))
9130 POKE can+12,(INT (d/256))
9140 RETURN
9148
9149
9150 REM  setta n. dati a 512
9160 LET can=23844
9170 POKE (can+69),0: POKE (can+
70),2
9174 RETURN
9175
9176
9200 REM   CONTROLLA  LUNGHEZZA
9210 LET CAN=23844
9220 LET chbyte=(256*PEEK (12+ca
n))+PEEK (11+can)
9240 IF ((chbyte+(LEN d$))>512)
THEN  PRINT "illegal quantity ER
ROR": LET d$=""
9250 RETURN
9298
9299
9300 REM scrive nel buffer
9310 LET can=23844
9320 LET chbyte=(256*PEEK (12+ca
n))+PEEK (11+can)
9330 LET ld=LEN d$
9340 LET start=can+82+chbyte
9350 FOR c=start+1 TO start+ld
9360 LET d=CODE d$(c-start)
9370 POKE c-1,d
9380 NEXT c
9385 LET sum=chbyte+ld
9390 POKE can+11,sum-256*(INT (s
um/256))
9395 POKE can+12,INT (sum/256)
```

```
9396 RETURN
9398
9399
9400 REM  PROC. DI  WRITE
9410 GO SUB 8050: REM  SCRITTURA
9420 GO SUB 9050: REM gen 1
9430 GO SUB 8100: REM pos. reco
9440 GO SUB 9000: REM lib. rec.
9450 GO SUB 8200: REM go print
9460 GO SUB 8079: REM lettura
9470 RETURN
```

```
   0>REM ***  PROGRAMMA UNO ***
   1 LET a$="####################
##################################
##################################
#################"
   2 OPEN #4;"m";1;"file"
   3 FOR x=1 TO 15: PRINT #4;a$;
   4 NEXT x
   5 CLOSE #4
   7
   8
   9 OPEN #4;"m";1;"file"
  10 FOR x=1 TO 100
  15 PRINT INKEY$#4;
  20 NEXT x
  30
  31
  60 PRINT ';"SCRIVI QUELLO CHE
VUOI E POI    PREMI ENTER DUE VO
LTE"
  70 INPUT "scrivi: ";d$: IF d$=
"" THEN  GO TO 80
  74 GO SUB 9200
  76 GO SUB 9300
  78 GO TO 70
  80 GO SUB 9400: REM  write
  90 CLOSE #4
 100  OPEN #4;"m";1;"file"
 110 PRINT INKEY$#4;
 120 GO TO 110
```

```
   0>REM *** PROGRAMMA DUE ***
   1 LET a$="####################
##################################
##################################
#################"
   2  OPEN #4;"m";1;"file"
   3 FOR x=1 TO 15: PRINT #4;a$;
   4 NEXT x
   5 CLOSE #4
   7
   8
   9  OPEN #4;"m";1;"file"
  10 PRINT "PREMI s QUANDO TI VU
OI FERMARE "
  12 PRINT INKEY$#4;
  15 IF INKEY$="s" THEN  GO TO 6
0
  17 PAUSE 3
  20 GO TO 12
  30
  31
  60 PRINT '"SCRIVI QUELLO CHE V
UOI E POI    PREMI ENTER DUE VOL
TE"
  70 INPUT "scrivi: ";d$: IF d$=
"" THEN  GO TO 80
  74 GO SUB 9200
  76 GO SUB 9300
  78 GO TO 70
  80 GO SUB 9400: REM write
  90 CLOSE #4
 100 GO TO 9
```

```
   0>REM *** PROGRAMMA TRE ***
   1 LET a$="####################
##################################
##################################
#################": LET can=2384
4
   2 OPEN #4;"m";1;"file"
   3 FOR x=1 TO 15: PRINT #4;a$;
   4 NEXT x
   5 CLOSE #4
   7
   8
  10 INPUT "vuoi modificare o in
serire un   nuovo blocco ? (m/i)
";d$
  15 IF d$="m" THEN  GO TO 100
  20 OPEN #4;"m";1;"file": GO SU
B 8000: REM  mem. settori liberi
  25 IF PEEK (can+67)=2 THEN  GO
TO 27
  26 GO SUB 8250: GO TO 25
  28 GO SUB 9150: REM  sett. n.
  29 GO SUB 8170: REM canc. EOF
  30 GO SUB 9400: REM  write
  40 GO SUB 9150: REM  sett. n.
  50 GO SUB 8150: REM ins . EOF
  55 LET X=1
  60 IF P(X)=0 THEN  POKE (can+4
1),x: GO TO 160
  70 LET x=x+1: GO TO 60
  98
  99
 100 OPEN #4;"m";1;"file"
 110 PRINT "PREMI s QUANDO TI VU
OI FERMARE "
 120 PRINT INKEY$#4;
 125 PAUSE 2
 130 IF INKEY$="s" THEN  GO TO 1
60
 150 GO TO 120
 158
 159
 160 PRINT ';"SCRIVI QUELLO CHE
VUOI E POI    PREMI ENTER DUE VO
LTE"
```

```
 170 INPUT "scrivi: ";d$: IF d$=
"" THEN  GO TO 180
 174 GO SUB 9200
 176 GO SUB 9300
 178 GO TO 170
 180 GO SUB 9400: REM  write
 190 CLOSE #4
 200 GO TO 9
```

Polinomi

```
  10 REM
            EQUAZIONI

        by Giuseppe Alcini
                &
          Stefano Salsano


  40 REM inizializzazioni
  41 DIM s(100): LET p=1
  42 DIM t(100): LET q=1
  45 DIM a(100)
  47 DIM b(100)
  50 DIM u(100): LET u=1
 100 PRINT "********************
********************  EQUAZIONI
 ********************    by STIG
 ******************************
************"
 105 PRINT
 107 REM inserimento dati
 110 INPUT "grado equazione ";gr
: IF gr<1 THEN  GO TO 110
 120 LET p2=1
 130 INPUT AT 1,0;"x";AT 0,1;VAL
 "(gr+1)-p2";AT 1,3;"= ";a(p2)
 135 PRINT "x^";VAL "(gr+1)-p2";
"= ";a(p2)
 139 LET p2=p2+1
 140 IF p2<=gr THEN  GO TO 130
 145 INPUT "termine noto= ";a(p2
)
 150 PRINT "termine noto= ";a(p2
)
 153 LET tn=a(p2)
 154 IF tn=0 THEN  LET p2=p2-1:
GO TO 153
 159 REM divisori t.noto
 160 FOR f=ABS tn TO -ABS tn STE
P -1
 162 IF f=0 THEN  NEXT f
 163 IF tn/f=INT (tn/f) THEN  LE
T s(p)=f: LET p=p+1
 165 NEXT f
 166 REM divisori 1 coeff.
 167 FOR f=ABS a(1) TO 1 STEP -1
 169 IF a(1)/f=INT (a(1)/f) THEN
  LET t(q)=f: LET q=q+1
 171 NEXT f
 175 REM calcolo soluzioni
 176 PRINT : PRINT "************
**********************"
 177 PRINT
 178 PRINT "soluzioni: ";
 179 FOR n=1 TO p-1: FOR o=1 TO
q-1: LET r=s(n)/t(o)
 180 FOR c=1 TO u-1: IF r=u(c) T
HEN  GO TO 204
 181 NEXT c
 183 REM ruffini
 184 FOR i=2 TO p2
 185 LET b(1)=a(1)
 190 LET b(i)=(b(i-1)*r)+a(i)
 200 NEXT i
 202 IF ABS (b(i-1))<0.000001 TH
EN  GO SUB 250
 203 IF u>gr THEN  GO TO 219
 204 NEXT o: NEXT n
 208 IF p2<=gr THEN  PRINT TAB 1
1;0: GO TO 219
 210 IF u=1 THEN  PRINT TAB 11;"
 nessuna "
 219 PRINT : PRINT "************
**********************"
 220 PAUSE 100: PRINT #1;"  prem
i un tasto": PAUSE 0: IF INKEY$=
"e" THEN  STOP
 222 RUN
 250 REM semplif. & stampa
 260 LET u(u)=r: LET u=u+1
 270 IF r=INT (r) THEN  PRINT TA
B 11;r: RETURN
 280 LET a=s(n): LET b=t(o)
 285 IF a=1 THEN  GO TO 320
 290 FOR i=(ABS (a*(a<b))+ABS (b
*(b<a))) TO 2 STEP -1
 300 IF (a/i=INT (a/i)) AND (b/i
=INT (b/i)) THEN  PRINT TAB 11;a
/i;"/";b/i: RETURN
 310 NEXT i
 320 PRINT TAB 11;a;"/";b
 330 RETURN
```

Grafici polari

```
   1 REM   GRAFICI DI FUNZIONI
           IN FORMA POLARE
         by Stefano Pizzeghello
              #(C)  1985 #
   9 REM
  10 REM input dati
  12 LET flag=0
  15 PRINT TAB 7; INVERSE 1; BRI
GHT 1;"I N P U T  D A T I"
  20 PRINT AT 4,0; FLASH 1;"1";
FLASH 0;" - Funzione f(";CHR$ 14
4;"): "
  25 INPUT "(T sostituisce ";CHR
$ 144;") "; LINE f$: IF f$="" TH
EN  BEEP 1,-15: PRINT #0; FLASH
```

```
1;"Funzione non definita": PAUSE
 150: GO TO 25
  26 PRINT TAB 4;f$'
  30 IF flag=1 THEN  GO TO 100
  40 PRINT AT 8,0; FLASH 1;"2";
FLASH 0;" - Intervallo dell'ango
lo ";CHR$ 144;": "'TAB 4;"estrem
o sinistro: ";: INPUT  LINE a$:
IF a$="" THEN  GO TO 40
  41 LET t1=VAL a$: PRINT a$'
  42 PRINT AT 10,4;"estremo dest
ro  : ";: INPUT  LINE b$: IF b$=
"" THEN  GO TO 42
  45 LET t2=VAL b$
  50 IF t2<=t1 THEN  BEEP 1,-15:
 PRINT #0; FLASH 1;"Il secondo e
stremo deve essere  maggiore del
 primo: reinserisci ": PAUSE 150
: GO TO 42
  55 PRINT b$
  58 IF flag=1 THEN  GO TO 100
  60 PRINT AT 12,0; FLASH 1;"3";
 FLASH 0;" - Shift dell'angolo:
";
  62 INPUT "(Valore consigliato:
 .07)"' LINE s$: IF s$="" THEN
GO TO 62
  65 LET st=VAL s$: IF st<=PI/60
 OR st>=PI/30 THEN  BEEP 1,-15:
PRINT #0; FLASH 1;"Deve essere c
ompreso tra"'"PI/60 e PI/30
       ": PAUSE 150: GO TO 62
  75 PRINT s$
  78 IF flag=1 THEN  GO TO 100
  80 PRINT AT 16,0; FLASH 1;"4";
 FLASH 0;" - Intervallo ascissa:
 ";
  85 INPUT "Inserire solo il val
ore positivo";asc: IF asc<=0 THE
N  BEEP 1,-15: PRINT #0; FLASH 1
;"Deve essere positivo!": PAUSE
150: GO TO 80
  86 PRINT asc
  89 REM calcolo delle dimensio-
        ni dello schermo
  90 LET ord=asc*175/255
  95 PRINT AT 17,4;"asse x","ass
e y"'TAB 4;-asc,-ord'TAB 4;asc,o
rd
 100 PRINT #0;"Confermi i dati?
s/n": PAUSE 0: INPUT ;: IF INKEY
$="S" THEN  GO TO 1000
 200 REM correzione dati
 210 PRINT #0;"Correggi tutto? s
/n": PAUSE 0: INPUT ;: IF INKEY$
="S" THEN  RUN
 220 LET flag=1: PRINT #0;"Inser
isci il n. che corrisponde al da
to errato": PAUSE 0: LET r$=INKE
Y$: INPUT ;
 225 IF r$<"1" OR r$>"4" OR r$="
" THEN  GO TO 220
 230 GO SUB 8500: GO TO 20*VAL r
$
1000 REM inizio elaborazione
1001 CLS : PLOT 0,87: DRAW 255,0
: PLOT 127,0: DRAW 0,175
1002 IF asc<=1 THEN  GO TO 1019
1003 LET unpix=INT (128/asc)
1004 FOR f=127+unpix TO 255 STEP
 unpix: PLOT f,84: DRAW 0,6: NEX
T f
1005 FOR f=127-unpix TO 0 STEP -
unpix: PLOT f,84: DRAW 0,6: NEXT
 f
1006 FOR f=87+unpix TO 175 STEP
unpix: PLOT 124,f: DRAW 6,0: NEX
T f
1007 FOR f=87-unpix TO 0 STEP -u
npix: PLOT 124,f: DRAW 6,0: NEXT
 f
1008 FOR f=127+unpix/2 TO 255 ST
EP unpix: PLOT f,86: DRAW 0,2: N
EXT f
1009 FOR f=127-unpix/2 TO 0 STEP
 -unpix: PLOT f,86: DRAW 0,2: NE
XT f
1010 FOR f=87+unpix/2 TO 175 STE
P unpix: PLOT 126,f: DRAW 2,0: N
EXT f
1011 FOR f=87-unpix/2 TO 0 STEP
-unpix: PLOT 126,f: DRAW 2,0: NE
XT f
1019 PRINT AT 0,17;"Y";AT 12,31;
"X": LET mol=127/asc
1020 REM tracciamento
1025 FOR f=t1 TO t2 STEP st
1030 LET x=127+mol*COS f*FN f(f)
1045 LET y=87+mol*SIN f*FN f(f)
1050 IF x<0 OR x>255 OR y<0 OR y
>175 THEN  NEXT f: IF f>t2 THEN
 GO TO 1100
1060 PLOT x,y
1080 NEXT f
1100 BEEP .5,12
2000 REM menu'
2010 PRINT #0; FLASH 1;"C"; FLAS
H 0;"opy "; FLASH 1;"M"; FLASH 0
;"odifica rapporto "; FLASH 1;"R
"; FLASH 0;"un "; FLASH 1;"S"; F
LASH 0;"top ": PAUSE 0: LET o$=I
NKEY$: INPUT ;
2015 IF o$<>"C" AND o$<>"M" AND
o$<>"R" AND o$<>"S" THEN  BEEP 1
,-15: GO TO 2010
2020 GO TO (7000 AND o$="C")+(25
00 AND o$="M")+(9995 AND o$="S")
+(9015 AND o$="R")
2500 REM scelta rapporto di
          riproduzione
2505 INPUT "Inserisci il rapport
o ";rap: IF rap<=0 THEN  BEEP 1,
-15: PRINT #0; FLASH 1;"Deve ess
ere positivo!": PAUSE 150: GO TO
 2505
2520 PRINT #0; FLASH 1;"I"; FLAS
H 0;"ngrandimento "; FLASH 1;"R"
; FLASH 0;"iduzione": PAUSE 0: L
```

```
ti o$=INKEY$: INPUT ;: IF o$<>"I
" AND o$<>"R" THEN  BEEP 1,-15:
GO TO 2520
2530 IF o$="I" THEN  LET rap=1/r
ap
2550 REM aggiornamento
2600 LET asc=asc*rap: LET ord=as
c*175/255: LET st=st*rap
2610 GO TO 1000
7000 REM copy dello schermo
7005 PRINT #0; FLASH 1;"N"; FLAS
H 0;"astro o "; FLASH 1;"S"; FLA
SH 0;"tampante?": PAUSE 0: INPUT
 ;: IF INKEY$="N" THEN  GO TO 71
00
7010 LPRINT TAB 5; INVERSE 1;"GR
AFICO DELLA FUNZIONE": LPRINT
7012 LPRINT "f(";CHR$ 144;")=";f
$: LPRINT
7013 LPRINT "Intervallo ";CHR$ 1
44;":"'a$,b$
7015 LPRINT "Intervallo ascissa:
"'-asc,asc
7020 LPRINT "Intervallo ordinata
:"'-ord,ord
7025 LPRINT "Shift angolo ";CHR$
 144;": ";st: LPRINT
7030 COPY : PAUSE 150: BEEP .5,1
2: GO TO 2000
7100 INPUT "Nome dello schermo?"
' LINE n$: IF n$="" OR LEN n$>10
 THEN  BEEP 1,-15: GO TO 7100
7110 SAVE n$SCREEN$ : BEEP .5,12
: POKE 23658,8: GO TO 2000
8500 REM subroutine "cancella"
8520 PRINT AT 4*VAL r$,0;
8530 FOR f=0 TO 127: PRINT CHR$
32;: NEXT f
8535 RETURN
9000 REM presentazione
9005 BEEP .5,15: BEEP 1,20: POKE
 23609,20: BORDER 0: PAPER 0: IN
K 7: CLS : PRINT AT 8,9; INVERSE
 1; BRIGHT 1;"F U N Z I O N I";A
T 10,9;"IN FORMA POLARE";AT 18,3
; INVERSE 0; BRIGHT 0;"? Stefano
 Pizzeghello 1985"
9010 REM caricamento carattere
          grafico; def funzione
9011 PAUSE 300: RESTORE 9020: FO
R f=0 TO 7: READ d: POKE USR "a"
+f,d: NEXT f: POKE 23658,8
9015 RUN : DEF FN f(t)=VAL f$
9020 DATA 0,48,74,124,72,72,48,0
9990 CLEAR : SAVE "grafpolari" L
INE 9000
9995 STOP
```

Integrali doppi

```
1 REM    PROGRAMMA INTEGRALI
         DOPPI
                1985 (C) by PIERLUIGI
                     PACCAGNELLA
  10 POKE 23609,30: DIM k(12): D
IM w(12)
  20 DATA .1252334085,.367831498
9,.5873179542,.7699026741,.90411
72563,.9815606342,.2491470458,.2
334925365,.2031674267,.160078328
5,.1069393259,.0471753363
  30 FOR i=1 TO 6: READ k(i): LE
T k(i+6)=-k(i): NEXT i
  40 FOR i=1 TO 6: READ w(i): LE
T w(i+6)=w(i): NEXT i
  50 BRIGHT 1: BORDER 1: PAPER 1
: INK 7: CLS
  60 PRINT  BRIGHT 1;
   " PROGRAMMA PER CALCOLO INTEG
RALI DOPPI, NORMALI RISPETTO
     ALL'ASSE X   (se fosse norm
ale  rispetto all'asse Y, invert
ire  X con Y nelle funzioni);"
  70 PRINT ''
 " L'INTEGRALE E' CALCOLATO DA A
 A B RISPETTO A X, E DA finf A
    fsup RISPETTO A Y
    (per dire infinito impostare
+i   oppure -i)"
  80 INPUT "A="; LINE a$,"B="; L
INE b$''"finf="; LINE i$''"fsup=
"; LINE s$''"F(x,y)="; LINE f$
  90 LET z=1: IF b$="+i" OR a$="
-i" THEN  GO SUB 160: GO TO 150
 100 LET a=VAL a$: LET b=VAL b$:
 110 LET somma=0: FOR i=1 TO 12:
 LET x=(b-a)*.5*k(i)+(b+a)*.5: L
ET finf=VAL i$: LET fsup=VAL s$
 120 GO SUB 230
 130 LET somma=somma+sum*w(i): N
EXT i
 140 LET somma=somma*(b-a)*.5
 150 PRINT ''''"IL RISULTATO E':
";somma: STOP
 160 IF a$="-i" AND b$="+i" THEN
  GO TO 220
 170 IF a$="-i" THEN  LET a$="-"
+b$: LET z=-1
 180 LET a=VAL a$: LET somma=0:
FOR i=1 TO 12: LET x=a-1+2/(1+k(
i)): LET x=x*z: LET finf=VAL i$:
 LET fsup=VAL s$
 190 GO SUB 230
 200 LET somma=somma+sum*w(i)/((
1+k(i))^2): NEXT i
 210 LET somma=somma*2: RETURN
 220 LET a$="0": GO SUB 160: LET
 ss=somma: LET a$="-i": LET b$="
0": GO SUB 160: LET somma=somma+
ss: GO TO 150
 230 LET sum=0: FOR j=1 TO 12: L
ET y=(fsup-finf)*.5*k(j)+(fsup+f
inf)*.5
 240 LET x=x*z: LET sum=sum+w(j)
*(VAL f$): NEXT j
```

```
250 LET sum=sum*(fsup-finf)*.5
260 RETURN
```

LIST controllato

```
  1 REM     uso di "extend"
      SENZA l'interfaccia 1

  2 DEF FN l(s,x,y)=USR st: DEF
 FN c()=USR (st+68):                R
EM
      LA RAMTOP DEVE ESSERE POST
A
                A  st-1
    CON    " CLEAR (st-1) ",
    dove "st" deve essere
         MINORE di 65294

  3 REM     Per il listato dalla
    linea "a" alla linea "b"
    sul canale "c" dare :
   RANDOMIZE  FN l(c,a,b) :
( c=2 per il listato sul VIDEO ,
   c=3 sulla stampante    ) .

    Per il COPY a 24 linee dare
  RANDOMIZE USR c()  .
   5 REM  ===>>>> EXTEND.S
   6 REM
  10 IF PEEK 23733<>255 AND PEEK
 23733<>127 THEN  STOP
  20 LET st=65294-32768*(PEEK 23
733=127)
  30 CLEAR st-1: LET st=PEEK 237
30+256*PEEK 23731+1
  40 LET a$="2a0b5c7cb52002cf090
6042310fd7ee5cd0116e10608231 0fd4
e237ee63f47c506072310fdc15e237ee
63f57d5606922495ccd6e19c103c5c1c
d8019d0c5cd5518d718f4f306c0c3af0
e"
  60 IF LEN a$<>148 THEN  GO TO
9000
  70 LET ck1=0: LET ck2=0
  80 FOR i=0 TO 73
  90 LET h=CODE a$(2*i+1)-48-39*
(a$(2*i+1)>"9")
 100 LET l=CODE a$(2*i+2)-48-39*
(a$(2*i+2)>"9")
 110 LET v=h*16+l
 120 LET ck1=ck1+l+h: LET ck2=ck
2+(i+1)*v
 130 POKE st+i,v
 140 PRINT #0;AT 1,0;i: NEXT i
 150 IF ck1<>989 OR ck2<>334360
THEN  GO TO 9000
 160 INPUT "Vuoi salvare i codic
i ? "; LINE a$: IF a$="s" OR a$=
"S" THEN  SAVE "CODE EXTEND"CODE
 st,74: VERIFY "CODE EXTEND"CODE

 170 PRINT "EXTEND e' in memoria
 a partire  da ";st
 190 STOP
9000 PRINT "ERRORE DI CARICAMENT
O DATI"''"CONTROLLA MEGLIO"
9010 STOP
```

```
FF0E  2A0B5C        LD   HL,(#5C0B)
FF11  7C            LD   A,H
FF12  B5            OR   L
FF13  2002          JR   NZ,LFF17
FF15  CF            RST  8
FF16  09            DEFB #09
FF17  0604    LFF17 LD   B,#04
FF19  23      LFF19 INC  HL
FF1A  10FD          DJNZ LFF19
FF1C  7E            LD   A,(HL)
FF1D  E5            PUSH HL
FF1E  CD0116        CALL #1601
FF21  E1            POP  HL
FF22  0608          LD   B,#08
FF24  23      LFF24 INC  HL
FF25  10FD          DJNZ LFF24
FF27  4E            LD   C,(HL)
FF28  23            INC  HL
FF29  7E            LD   A,(HL)
FF2A  E63F          AND  #3F
FF2C  47            LD   B,A
FF2D  C5            PUSH BC
FF2E  0607          LD   B,#07
FF30  23      LFF30 INC  HL
FF31  10FD          DJNZ LFF30
FF33  C1            POP  BC
FF34  5E            LD   E,(HL)
FF35  23            INC  HL
FF36  7E            LD   A,(HL)
FF37  E63F          AND  #3F
FF39  57            LD   D,A
FF3A  D5            PUSH DE
FF3B  60            LD   H,B
FF3C  69            LD   L,C
FF3D  22495C        LD   (#5C49),HL
FF40  CD6E19        CALL #196E
FF43  C1            POP  BC
FF44  03            INC  BC
FF45  C5            PUSH BC
FF46  C1      LFF46 POP  BC
FF47  CD8019        CALL #1980
FF4A  D0            RET  NC
FF4B  C5            PUSH BC
FF4C  CD5518        CALL #1855
FF4F  D7            RST  #10
FF50  18F4          JR   LFF46
FF52  F3            DI
FF53  06C0          LD   B,#C0
FF55  C3AF0E        JP   #0EAF
```

```
  7 REM  ===>>> EXTEND.C
  8 REM
 10 IF PEEK 23733<>255 AND PEEK
23733<>127 THEN  STOP
 20 CLS #: LET st=65156-32768*(
PEEK 23733=127)
```

```
  30 CLEAR st-1: LET st=PEEK 237
30+256*PEEK 23731+1
  40 LET a$="c5cf31c1210c000922b
75cc9d71800fef0280ffee12807feff2
86ac3f0013e0318023e02fd360200d73
0252803d70116d72000d77020380ed71
800fe2c2804fe3b2021d72000fe2a281
cfecc2016d72000d7821ccdb705d7991
e78e63f47c50100001845e700d72000d
7821cfecc280afe3a"
  50 LET b$="2853fe0d284f18ead72
000d7821ccdb7051819d72000fe2320d
8d72000cdb70521c0500610f3d7b20ec
3c105d7991e78e63f47c5d7991e78e63
f676922495cd76e19c103c5c1d78019d
2c105c5d75518d7100018f0cdb705d79
91e78e63f47c518d5"
  60 LET c$=a$+b$: IF LEN c$<>42
4 THEN  GO TO 9000
  70 LET ck1=0: LET ck2=0
  80 FOR i=0 TO 211
  90 LET h=CODE c$(2*i+1)-48-39*
(c$(2*i+1)>"9")
 100 LET l=CODE c$(2*i+2)-48-39*
(c$(2*i+2)>"9")
 110 LET v=h*16+l
 120 LET ck1=ck1+l+h: LET ck2=ck
2+(i+1)*v
 130 POKE st+i,v
 140 PRINT #0;AT 1,0;i: NEXT i
 150 IF ck1<>2712 OR ck2<>252995
1 THEN  GO TO 9000
 160 INPUT "Vuoi salvare i codic
i ? "; LINE a$: IF a$="s" OR a$=
"S" THEN  SAVE "CODE EXTEND"CODE
 st,212: VERIFY "CODE EXTEND"COD
E
 170 PRINT "EXTEND e' in memoria
 a partire  da ";st
 180 RANDOMIZE USR st
 190 STOP
9000 PRINT "ERRORE DI CARICAMENT
O DATI"''''"CONTROLLA MEGLIO"
9010 STOP
```

```
FE84  C5            PUSH BC
FE85  CF            RST  8
FE86  31            DEFB "1"
FE87  C1            POP  BC
FE88  210C00        LD   HL,#000C
FE8B  09            ADD  HL,BC
FE8C  22B75C        LD   (#5CB7),HL
FE8F  C9            RET
FE90  D7            RST  #10
FE91  1800          DEFB #18,#00
FE93  FEF0          CP   #F0
FE95  280F          JR   Z,LFEA6
FE97  FEE1          CP   #E1
FE99  2807          JR   Z,LFEA2
FE9B  FEFF          CP   #FF
FE9D  286A          JR   Z,LFF09
FE9F  C3F001        JP   #01F0
FEA2  3E03    LFEA2 LD   A,#03
```

```
FEA4  1802          JR   LFEA8
FEA6  3E02    LFEA6 LD   A,#02
FEA8  FD360200 LFEA8 LD   (IY+2),#00
FEAC  D7            RST  #10
FEAD  3025          DEFB "0","%"
FEAF  2803          JR   Z,LFEB4
FEB1  D7            RST  #10
FEB2  0116          DEFB #01,#16
FEB4  D7      LFEB4 RST  #10
FEB5  2000          DEFB " ",#00
FEB7  D7            RST  #10
FEB8  7020          DEFB "p"," "
FEBA  380E          JR   C,LFECA
FEBC  D7            RST  #10
FEBD  1800          DEFB #18,#00
FEBF  FE2C          CP   #2C
FEC1  2804          JR   Z,LFEC7
FEC3  FE3B          CP   #3B
FEC5  2021          JR   NZ,LFEE8
FEC7  D7      LFEC7 RST  #10
FEC8  2000          DEFB " ",#00
FECA  FE2A    LFECA CP   #2A
FECC  281C          JR   Z,LFEEA
FECE  FECC          CP   #CC
FED0  2016          JR   NZ,LFEE8
FED2  D7            RST  #10
FED3  2000          DEFB " ",#00
FED5  D7            RST  #10
FED6  821C          DEFB #82,#1C
FED8  CDB705        CALL #05B7
FEDB  D7            RST  #10
FEDC  991E          DEFB #99,#1E
FEDE  78            LD   A,B
FEDF  E63F          AND  #3F
FEE1  47            LD   B,A
FEE2  C5            PUSH BC
FEE3  010000        LD   BC,#0000
FEE6  1845          JR   LFF2D
FEE8  E7      LFEE8 RST  #20
FEE9  00            DEFB #00
FEEA  D7      LFEEA RST  #10
FEEB  2000          DEFB " ",#00
FEED  D7            RST  #10
FEEE  821C          DEFB #82,#1C
FEF0  FECC          CP   #CC
FEF2  280A          JR   Z,LFEFE
FEF4  FE3A          CP   #3A
FEF6  2853          JR   Z,LFF4B
FEF8  FE0D          CP   #0D
FEFA  284F          JR   Z,LFF4B
FEFC  18EA          JR   LFEE8
FEFE  D7      LFEFE RST  #10
FEFF  2000          DEFB " ",#00
FF01  D7            RST  #10
FF02  821C          DEFB #82,#1C
FF04  CDB705        CALL #05B7
FF07  1819          JR   LFF22
FF09  D7      LFF09 RST  #10
FF0A  2000          DEFB " ",#00
FF0C  FE23          CP   #23
FF0E  20D8          JR   NZ,LFEE8
FF10  D7            RST  #10
```

```
FF11  2000          DEFB " ",#00
FF13  CDB705        CALL #05B7
FF16  21C050        LD   HL,#50C0
FF19  0610          LD   B,#10
FF1B  F3            DI
FF1C  D7            RST  #10
FF1D  B20E          DEFB #B2,#0E
FF1F  C3C105        JP   #05C1
FF22  D7       LFF22 RST  #10
FF23  991E          DEFB #99,#1E
FF25  78            LD   A,B
FF26  E63F          AND  #3F
FF28  47            LD   B,A
FF29  C5            PUSH BC
FF2A  D7            RST  #10
FF2B  991E          DEFB #99,#1E
FF2D  78       LFF2D LD   A,B
FF2E  E63F          AND  #3F
FF30  67            LD   H,A
FF31  69            LD   L,C
FF32  22495C        LD   (#5C49),HL
FF35  D7            RST  #10
FF36  6E19          DEFB "n",#19
FF38  C1            POP  BC
FF39  03            INC  BC
FF3A  C5            PUSH BC
FF3B  C1       LFF3B POP  BC
FF3C  D7            RST  #10
FF3D  8019          DEFB #80,#19
FF3F  D2C105        JP   NC,#05C1
FF42  C5            PUSH BC
FF43  D7            RST  #10
FF44  5518          DEFB "U",#18
FF46  D7            RST  #10
FF47  1000          DEFB #10,#00
FF49  18F0          JR   LFF3B
FF4B  CDB705   LFF4B CALL #05B7
FF4E  D7            RST  #10
FF4F  991E          DEFB #99,#1E
FF51  78            LD   A,B
FF52  E63F          AND  #3F
FF54  47            LD   B,A
FF55  C5            PUSH BC
FF56  18D5          JR   LFF2D
```

Bubble-s.

```
1000 REM     M/C BUBBLE SORT
1010 REM    GENERATORE CODICE
1020 REM BY LUIGI R. CALLEGARI
1030 REM ----------------------
1040 POKE 23609,22: BORDER 4
1050 PAPER 7: INK 9: BRIGHT 0
1060 OVER 0: CLEAR 26999
1070 DEF FN f(a$)=CODE a$-48-7*(
a$>"9")
1080 LET a=27000
1090 LET b=a: POKE 23692,255
1100 FOR j=1290 TO 1460 STEP 10
1110 LET ch=0
1120 READ a$: PRINT CHR$ 32;a$;"
 = ";
1130 IF a$="" THEN  GO TO 1260
1140 IF LEN a$=24 THEN  GO TO 11
80
1150 PRINT '''"ERRORE IN LINEA ";
 FLASH 1;j
1160 BEEP PI,PI
1170 STOP
1180 FOR k=1 TO 12: BORDER 7*RND
1190 LET x=FN f(a$(1))*16+FN f(a
$(2))
1200 LET a$=a$(3 TO )
1210 LET ch=ch+x: POKE b,x
1220 LET b=1+b: NEXT k
1230 PRINT ch: READ c
1240 IF c<>ch THEN  GO TO 1150
1250 NEXT j: BORDER 1
1260 BEEP 1,20: LET s$="m/c sort
"
1270 SAVE s$CODE a,(1+b-a)
1280 GO TO 9999
1290 DATA "F3CD6B0DAFCD01162A4B5
C7E",1306
1300 DATA "FE80CA7006FEC12806CDB
819",1609
1310 DATA "EB18F02323237EFE02C22
02A",1254
1320 DATA "235E2356ED53FE5A235E2
356",1164
1330 DATA "ED53FA5A2322F65A2AF65
AED",1680
1340 DATA "5BFA5A1922F85A2AFE5A2
B22",1291
1350 DATA "FC5A7DB42005FBCD6B0DC
9ED",1698
1360 DATA "4BFA5AED5BF65A2AF85A1
ABE",1675
1370 DATA "20410B132379B020F52AF
85A",1116
1380 DATA "ED5BFA5A1922F85A2AFC5
A2B",1492
1390 DATA "22FC5A7DB420D42AF65AE
D5B",1631
1400 DATA "FA5A1922F65A2AFE5AE52
B22",1427
1410 DATA "FE5A3E16D73E01D7AFD7C
1CD",1709
1420 DATA "1B1A3E20D7189538C4ED4
BFA",1349
1430 DATA "5AED5BF65A2AF85A7EF51
A77",1650
1440 DATA "E607D3FEF1120B231379B
020",1355
1450 DATA "EF18A600000000000000
000",0429
1460 DATA ""
```

```
5B00  F3            DI
5B01  CD6B0D        CALL #0D6B
5B04  AF            XOR  A
```

```
5B05  CD0116          CALL #1601
5B08  2A4B5C          LD   HL,(#5C4B)
5B0B  7E        L5B0B LD   A,(HL)
5B0C  FE80            CP   #80
5B0E  CA7006          JP   Z,#0670
5B11  FEC1            CP   #C1
5B13  2806            JR   Z,L5B1B
5B15  CDB819          CALL #19B8
5B18  EB              EX   DE,HL
5B19  18F0            JR   L5B0B
5B1B  23        L5B1B INC  HL
5B1C  23              INC  HL
5B1D  23              INC  HL
5B1E  7E              LD   A,(HL)
5B1F  FE02            CP   #02
5B21  C2202A          JP   NZ,#2A20
5B24  23              INC  HL
5B25  5E              LD   E,(HL)
5B26  23              INC  HL
5B27  56              LD   D,(HL)
5B28  ED53FE5A        LD   (#5AFE),DE
582C  23              INC  HL
582D  5E              LD   E,(HL)
582E  23              INC  HL
582F  56              LD   D,(HL)
5830  ED53FA5A        LD   (#5AFA),DE
5834  23              INC  HL
5835  22F65A          LD   (#5AF6),HL
5838  2AF65A    L5B38 LD   HL,(#5AF6)
583B  ED5BFA5A        LD   DE,(#5AFA)
583F  19              ADD  HL,DE
5840  22F85A          LD   (#5AF8),HL
5843  2AFE5A          LD   HL,(#5AFE)
5846  2B              DEC  HL
5847  22FC5A          LD   (#5AFC),HL
584A  7D              LD   A,L
584B  B4              OR   H
584C  2005            JR   NZ,L5B53
584E  FB              EI
584F  CD6B0D          CALL #0D6B
5852  C9              RET
5853  ED4BFA5A L5B53 LD   BC,(#5AFA)
5857  ED5BF65A       LD   DE,(#5AF6)
585B  2AF85A          LD   HL,(#5AF8)
585E  1A        L5B5E LD   A,(DE)
585F  BE              CP   (HL)
5860  2041            JR   NZ,L5BA3
5862  0B              DEC  BC
5863  13              INC  DE
5864  23              INC  HL
5865  79              LD   A,C
5866  B0              OR   B
5867  20F5            JR   NZ,L5B5E
5869  2AF85A    L5B69 LD   HL,(#5AF8)
586C  ED5BFA5A        LD   DE,(#5AFA)
5870  19              ADD  HL,DE
5871  22F85A          LD   (#5AF8),HL
5874  2AFC5A          LD   HL,(#5AFC)
5877  2B              DEC  HL
5878  22FC5A          LD   (#5AFC),HL
587B  7D              LD   A,L
587C  B4              OR   H
587D  20D4            JR   NZ,L5B53
587F  2AF65A          LD   HL,(#5AF6)
5882  ED5BFA5A        LD   DE,(#5AFA)
5886  19              ADD  HL,DE
5887  22F65A          LD   (#5AF6),HL
588A  2AFE5A          LD   HL,(#5AFE)
588D  E5              PUSH HL
588E  2B              DEC  HL
588F  22FE5A          LD   (#5AFE),HL

5B92  3E16            LD   A,#16
5B94  D7              RST  #10
5B95  3E01            LD   A,#01
5B97  D7              RST  #10
5B98  AF              XOR  A
5B99  D7              RST  #10
5B9A  C1              POP  BC
5B9B  CD1B1A          CALL #1A1B
5B9E  3E20            LD   A,#20
5BA0  D7              RST  #10
5BA1  1895            JR   L5B38
5BA3  38C4      L5BA3 JR   C,L5B69
5BA5  ED4BFA5A        LD   BC,(#5AFA)
5BA9  ED5BF65A        LD   DE,(#5AF6)
5BAD  2AF85A          LD   HL,(#5AF8)
5BB0  7E        L5BB0 LD   A,(HL)
5BB1  F5              PUSH AF
5BB2  1A              LD   A,(DE)
5BB3  77              LD   (HL),A
5BB4  E607            AND  #07
5BB6  D3FE            OUT  (#FE),A
5BB8  F1              POP  AF
5BB9  12              LD   (DE),A
5BBA  0B              DEC  BC
5BBB  23              INC  HL
5BBC  13              INC  DE
5BBD  79              LD   A,C
5BBE  B0              OR   B
5BBF  20EF            JR   NZ,L5BB0
5BC1  18A6            JR   L5B69
```

```
1000 REM DEMO PROGRAMMA M/C SORT
1010 REM (C)1985 LUIGI CALLEGARI
1020 REM ----------------------
1021 INK 9: PAPER 7: BRIGHT 0: O
VER 0: INVERSE 0
1025 CLEAR : RANDOMIZE
1030 CLS : INPUT "Quanti element
i casuali ? ";n
1040 PRINT '"     RIORDINO DI ";
n;" ELEMENTI"
1050 PRINT '"     LUNGHI 2 CAR. -
 CASUALI"
1060 DIM a$(n,2): PRINT  FLASH 1
;'''"GENERAZIONE..."
1070 FOR j=1 TO n: PRINT AT 8,15
;j
1080 LET a$(j)=CHR$ (64+RND*9)+C
HR$ (64+RND*25)
1090 NEXT j
1100 BORDER 2: CLS : PRINT "3 se
condi e parte il riordino."
1110 PAUSE 150
1130 RANDOMIZE USR 23296
1140 BEEP .5,10: BEEP .5,13
1150 FOR j=1 TO n
1160 PRINT a$(j);CHR$ 143;: NEXT
 j
1170 PRINT : LIST
1180 STOP
1190 SAVE "DEMOSORTMC"CODE 23296
,PEEK 23653+256*PEEK 23654-23250
1200 RUN
```

Boxe

```
  1 GO SUB 9300
  2 LET R=1: LET A=0: LET B=0
  4 BORDER 0: PAPER 6: INK 0: CLS
  5 DIM z$(3): DIM w$(3): INPUT "tuo
none ?";z$: INPUT "nome spectrum ?";
w$
  6 PRINT "INSERISCI LA VELOCITA' DU
E VOLTE"
  7 IF INKEY$="1" THEN  LET v=3: LET
 q$="MASSIMI": GO TO 13
  8 IF INKEY$="2" THEN  LET v=4: LET
 q$="MELTER": GO TO 13
  9 IF INKEY$="3" THEN  LET v=5: LET
 q$="PIUMA": GO TO 13
 10 GO TO 7
 13 PRINT "pesi :";q$: PAUSE 0: PAUS
E 0
 14 IF INKEY$="3" THEN  LET s=3: GO
 TO 9000
 15 IF INKEY$="1" THEN  LET s=1: GO
 TO 9000
 16 IF INKEY$="2" THEN  LET s=2: GO
 TO 9000
 17 GO TO 14
 503 LET teapo=90
 510 LET x=14: LET y=10: LET j=14: LE
T k=20
 511 PAUSE 100: BEEP 1,20
 515 LET f$=" ": LET g$=" ": LET h$=
" ": LET i$=" ": LET I$=" "
 520 LET d$=CHR$ 129+CHR$ 130+CHR$ 13
1: LET e$=CHR$ 131+CHR$ 129+CHR$ 130:
 LET x$=CHR$ 144: LET j$=CHR$ 147: LE
T y$=CHR$ 145+CHR$ 130: LET k$=CHR$ 1
29+CHR$ 148: LET a$=CHR$ 154: LET b$=
CHR$ 150: LET c$=CHR$ 151+CHR$ 152+CH
R$ 153
 521 PRINT AT 2,21; PAPER 0;teapo
 522 LET teapo=teapo-1
 525 IF teapo<0 THEN  GO TO 9500
 530 PRINT AT x,y+1;" "
 570 IF x=12 THEN  LET f$=" ": LET g$
=" ": LET h$=" ": LET i$=" "
 571 IF j=12 THEN  LET a$=" ": LET
n$=" ": LET o$=" ": LET p$=" "
 575 IF x=13 THEN  LET f$=" ": LET g$
=" ": LET h$=" ": LET i$=" "
 576 IF j=13 THEN  LET a$=" ": LET
n$=" ": LET o$=" ": LET p$=" "
 578 IF x=14 THEN  LET f$=" ": LET g$
=" ": LET h$=" ": LET i$=" "
 579 IF j=14 THEN  LET a$=" ": LET
n$=" ": LET o$=" ": LET p$=" "
 580 IF x=15 THEN  LET f$=" ": LET g$
=" ": LET h$=" ": LET i$=" "
 581 IF j=15 THEN  LET a$=" ": LET
n$=" ": LET o$=" ": LET p$=" "
 589 IF INKEY$="q" AND x>12 THEN  PRI
NT AT x,y; PAPER 8;f$;AT x+1,y;g$;AT
 x+2,y;h$;AT x+3,y-1;i$;AT x+4,y-1;"
 ": LET x=x-1
 590 IF INKEY$="a" AND x<15 THEN  PRI
NT AT x,y; PAPER 8;f$;AT x+1,y;g$;AT
 x+2,y;h$;AT x+3,y-1;i$;AT x+4,y-1;"
 ": LET x=x+1
 595 LET m=INT (RND*v): IF m=0 THEN
  PRINT AT j,k-2; PAPER 8;a$;AT j+1,k
-1;n$;AT j+2,k;o$;AT j+3,k-1;p$;AT j+
4,k-1;"  ": LET k=k+INT (RND*3)-1
 598 LET vi=INT (RND*v): IF vi=0 THEN
  PRINT AT j,k-2; PAPER 8;a$;AT j+1,k
-1;n$;AT j+2,k;o$;AT j+3,k-1;p$;AT j+
4,k-1;"  ": LET j=j+INT (RND*3)-1
 610 IF INKEY$="m" AND y<k-3 THEN  PR
INT AT x,y; PAPER 8;f$;AT x+1,y;g$;AT
 x+2,y;h$;AT x+3,y-1;i$;AT x+4,y-1;"
 ": LET y=y+1
 640 IF INKEY$="n" AND y>9 THEN  PRIN
T AT x,y; PAPER 8;f$;AT x+1,y;g$;AT x
+2,y;h$;AT x+3,y-1;i$;AT x+4,y-1;"
": LET y=y-1
 700 IF INKEY$="w" THEN  LET x$=CHR$
144+CHR$ 148+CHR$ 146
2500 LET pu=INT (RND*s): IF x=j AND p
u=0 AND m<>0 AND vi<>0 THEN  LET j$=
CHR$ 149+CHR$ 148+CHR$ 147: PRINT AT
j,k-2; INK 0;j$: LET j$=CHR$ 147
2600 IF x=j AND k=y+3 AND x$=CHR$ 144
+CHR$ 148+CHR$ 146 THEN  BEEP .005,0:
 LET a=a+2
2610 IF x=j AND k=y+3 AND pu=0 THEN
BEEP .005,0: LET b=b+2
2620 IF x=j AND k=y+3 AND pu=0 AND x$
=CHR$ 144+CHR$ 148+CHR$ 146 THEN  LET
 b=b-1: LET a=a-1
2700 IF j<12 THEN  LET j=12
2705 IF j>14 THEN  LET j=14
2710 IF k>22 THEN  LET k=22
2720 IF k<y+3 THEN  LET k=y+3
2800 IF b=75 OR b=76 THEN  GO TO 9750
2805 IF a=75 OR a=76 THEN  GO TO 9800
2900 PRINT AT x,y; INK 2; PAPER 8;x$;
AT x+1,y;y$;AT x+2,y;a$;AT x+3,y-1;c$
;AT x+4,y-1;d$
2910 PRINT AT j,k; INK 0; PAPER 8;j$;
AT j+1,k-1;k$;AT j+2,k;b$;AT j+3,k-1;
c$;AT j+4,k-1;e$
3000 GO TO 520
9000 LET r$="
          ": PRINT "velocita':";s: PAUS
E 100: INK 0: PRINT AT 0,0; INVERSE 1
;r$;r$
9010 PRINT AT 2,0; INVERSE 1;r$;r$;r$
9025 INK 7: PRINT AT 13,0;"----------
----------"
9030 PRINT AT 12,7;"
 "
9040 PRINT AT 14,7;"
 "
9045 LET r$="": FOR n=0 TO 17: LET r$
=r$+CHR$ 143: NEXT n
9050 PRINT AT 16,7; PAPER 7;r$
9055 PRINT AT 17,7; PAPER 7;r$
9058 PRINT AT 18,7; PAPER 7;r$
9060 PRINT AT 19,7; PAPER 7;r$
9062 PRINT AT 20,7; PAPER 7;r$
9063 PRINT AT 21,7; PAPER 7;r$
9070 FOR n=0 TO 56: PLOT N,0: DRAW 58
,47: NEXT N
9075 FOR n=255 TO 198 STEP -1: PLOT N
,0: DRAW -58,47: NEXT N
9080 PLOT 56,48: DRAW 0,28: PLOT 199,
48: DRAW 0,28
9090 PLOT 56,56: DRAW -58,-15
9095 PLOT 56,72: DRAW -58,15
9097 PLOT 199,56: DRAW 58,-15
9100 PLOT 199,72: DRAW 58,15
9105 LET s$=CHR$ 131
9110 PRINT AT 5,0; INK 0; INVERSE 1;"
          "; INVERSE 0;"       "; I
NVERSE 1;"          "; IN
VERSE 0;s$;"          ";s$; INVERSE
1;"          "
9120 PRINT AT 7,0; INK 0; INVERSE 1;"
          "; INVERSE 0;s$;"
 ";s$; INVERSE 1;"          "; IN
VERSE 0;s$;"          ";s$;
INVERSE 1;"          "; INVERSE 0;s$;"
          ";s$; INVERSE
1;"          "; INVERSE 0;s$;"
          ";s$; INVERSE 1;"  "
9130 PAPER 0: PRINT AT 0,1;"---------
----------"
9131 PRINT AT 1,1;"|"; INVERSE 1;"BOX
ER"; INVERSE 0;"| "; INVERSE 1;"1"; I
NVERSE 0;" | "; INVERSE 1;"2"; INVERS
E 0;" | "; INVERSE 1;"3"; INVERSE 0;"
|"; INVERSE 1;"TIME"; INVERSE 0;"|";
INVERSE 1;"FINALE"; INVERSE 0;"|"
9132 PRINT AT 2,1;"| "sz$(TO 3);" |
";AT 2,11;"| ";AT 2,15;"|  _|    |
  |"
9133 PRINT AT 3,1;"| ";w$(TO 3);" |
";AT 3,11;"| ";AT 3,15;"|   |"; INVER
SE 1;"TIME"; INVERSE 0;"|   |"
9140 PRINT AT 4,1;"----------
----------"
9160 PRINT AT 5,10; FLASH 1;"PESI ";q
$
9200 PAPER 6
9290 GO TO 500
9300 RESTORE 9400: FOR N=USR "a" TO U
SR "k"+7
9340 READ v: POKE n,v: NEXT n
9360 RESTORE : RETURN
9400 DATA 28,62,63,127,126,254,252,22
4,240,248,254,255,223,239,246,248,0,0
,63,255,247,199,0,0,56,124,252,254,12
6,127,63,7
9410 DATA 15,31,127,255,251,247,111,3
1,0,56,255,255,227,96,0,0,63,63,65,65
,129,255,231,199,207,207,135,135,131,
131,1,1,1,1,3,3,7,7,7,7,0,0,128,128,1
```

```
92,192,224,224
9420 DATA 252,252,130,130,129,255,231,227
9500 LET r=r+1: BEEP 2,3: BEEP .2,-2
9510 IF r=2 THEN  PRINT AT 2,9; PAPER 0;a;AT 3,9;b
9512 IF r=3 THEN  PRINT AT 2,13; PAPER 0;a;AT 3,13;b
9514 IF r>3 THEN  PRINT AT 2,17; PAPER 0;a;AT 3,17;b: GO
 TO 9700
9520 LET i=2: LET goto=9600
9521 FOR n=5 TO 21: PRINT AT n,0; INK 1; INVERSE 1;"
                          ": NEXT n
9524 PRINT AT 7,14; INK 0; PAPER 1;CHR$ 132+CHR$ 140+CHR
$ 140+CHR$ 136+"                          "+CHR$ 132; I
NVERSE 1;"    "; INVERSE 0;CHR$ 136;AT 10,13; PAPER 1;CH
R$ 138+"    "+CHR$ 133;AT 9,13;CHR$ 143+CHR$ 139+CHR$ 13
1+CHR$ 131+CHR$ 135+CHR$ 143;AT 11,13;CHR$ 138+"O O "+CH
R$ 133
9525 PRINT AT 12,13; PAPER 1; INK i; INVERSE 1;"      ";
 INVERSE 0;"                          "; INVERSE 1;"
  "; INVERSE 0;"                          ";CHR$ 133; IN
VERSE 1;"__ "+CHR$ 133
9528 PRINT AT 15,14; INK i; PAPER 1; INVERSE 1;" -  "; I
NVERSE 0;"                          "+CHR$ 135+CHR$ 14
3+CHR$ 139+CHR$ 138+CHR$ 136+"
"+CHR$ 132+CHR$ 142+CHR$ 140+CHR$ 140+CHR$ 143+CHR$ 143
9530 PRINT AT 18,8; PAPER 1; INK i; INVERSE 1;CHR$ 131+C
HR$ 131+"          "+CHR$ 129; INVERSE 0;"
   "+CHR$ 140; INVERSE 1;"                          "; INVERSE 0
;"          "; INVERSE 1;"                          "; INVER
SE 0;CHR$ 138+"          "; INVERSE 1;"    "
;  "+CHR$ 133
9535 GO TO goto
9600 PRINT AT 12,15; INK 3;CHR$ 143;AT 13,15;CHR$ 143
9605 IF b>18 THEN  PRINT AT 12,15; INK 0; PAPER 2;CHR$ 1
33;AT 13,15;CHR$ 143
9610 IF b>35 THEN  PRINT AT 11,16; INK 0;CHR$ 143;AT 13,
18; INK i; INVERSE 1;"\";AT 12,17;"\";AT 10,15;"\"
9620 IF b>50 THEN  PRINT AT 8,13; INK 7; PAPER 1; INVERS
E 1;CHR$ 139+"_/_";CHR$ 135;AT 9,13;"__/___"
9621 LET i=0: LET goto=9625
9622 PAUSE 0: PAUSE 0
9623 GO TO 9521
9625 PRINT AT 12,15; INK 7; PAPER 0;CHR$ 143;AT 13,15;CH
R$ 143;AT 11,14;"O";AT 11,16;"O"
9630 IF a>18 THEN  PRINT AT 12,15; INK 3; PAPER 0;CHR$ 1
33;AT 13,15;CHR$ 143
9635 IF a>35 THEN  PRINT AT 11,16; INK 2;CHR$ 143;AT 13,
18; INK i; INVERSE 1;"\";AT 12,17;"\";AT 10,15;"\"
9640 IF a>50 THEN  PRINT AT 8,13; INK 7; PAPER 1; INVERS
E 1;CHR$ 139+"_/_"+CHR$ 136;AT 9,13;"__/___"
9650 PAUSE 0: PAUSE 0
9651 FOR n=5 TO 21: PRINT AT n,0; PAPER 6;"
                    ": NEXT n
9660 GO TO 9025
9700 LET v$=w$
9701 IF a>b THEN  LET v$=z$
9706 PRINT AT 2,25; PAPER 0;v$( TO 3);" IS";AT 3,25; FLA
SH 1;"WINNER"
9709 IF INKEY$="s" THEN  RUN
9710 GO TO 9701
9750 LET per=0
9755 LET sal=INT (RND*(10+v)): IF sal=0 THEN  GO TO 2900
9756 BEEP .1,-30
9759 LET per=per+1
9760 IF per=10 THEN  GO TO 9765
9761 GO TO 9755
9765 PRINT AT 2,25; PAPER 0;z$( TO 3);" IS";AT 3,25;" K.
O. ": BEEP 3,0
9770 IF INKEY$="s" THEN  GO TO 9700
9771 GO TO 9770
9800 LET per=0
9805 LET sal=INT (RND*(10+v)): IF sal=0 THEN  GO TO 2900
9806 BEEP .1,30
9810 LET per=per+1
9811 IF per=10 THEN  GO TO 9820
9813 GO TO 9805
9820 PRINT AT 2,25; PAPER 0;w$( TO 3);" IS";AT 3,25;" K.
O. ": BEEP 3,0
9830 IF INKEY$="s" THEN  GO TO 9700
9840 GO TO 9830
9999 SAVE "BOXE" LINE 1
```

Spazio QL

Quella che segue e' una prima serie di PROCedure per simulare nuove funzioni sul QL. Per problemi di spazio, il commento apparira' sul prossimo numero.

```
100 DEFine PROCedure CHOICEWINDOW(a)
110  LOCal b,c,k,K$,x,y
120  IF a>2 THEN RETurn
130  b=40:c=20:x=0:y=0
140  MODE 4
150  OPEN#3,SCR_512X256A0X0
160  PAPER #3,0:CLS #3:PAPER #0,5:PAP
ER #1,3:PAPER #2,5:PAPER #a,7:INK #a,
0
170  REPeat all
180   WINDOW #a,b,c,32+x,16+y
190   CLS#3:CLS#0:CLS#1:CLS#2:CLS#a
200   PRINT #a;x;",";y\b;",";c
210   k=CODE(INKEY$(-1))
220   SELect ON k
230    ON k=196:x=x-1
240    ON k=204:x=x+1
250    ON k=197:x=x-4
260    ON k=198:x=x-8
270    ON k=205:x=x+4
280    ON k=206:x=x+8
290    ON k=212:y=y-1
300    ON k=213:y=y-4
310    ON k=214:y=y-8
320    ON k=220:y=y+1
330    ON k=221:y=y+4
340    ON k=222:y=y+8
350    ON k=192:b=b-1
```

```
360     ON k=193:b=b-4
370     ON k=194:b=b-8
380     ON k=200:b=b+1
390     ON k=201:b=b+4
400     ON k=202:b=b+8
410     ON k=208:c=c-1
420     ON k=209:c=c-4
430     ON k=210:c=c-8
440     ON k=216:c=c+1
450     ON k=217:c=c+4
460     ON k=218:c=c+8
470     ON k=10:EXIT all
480   END SELect
490   IF x)408 THEN x=408:ELSE IF x<0
THEN x=0:ELSE IF y>216 THEN y=216:ELS
E IF y<0 THEN y=0:ELSE IF b>448-x THE
N b=448-x:ELSE IF c>240-y THEN c=240-
y:ELSE IF c<20 THEN c=20:ELSE IF b<40
 THEN b=40
500   END REPeat all
510   END DEFine
```

```
100 DEFine FuNction MAX(a,b)
110   IF a)=b THEN
111     RETurn a
112   ELSE
120     RETurn b
121   END IF
130 END DEFine
```

```
100 DEFine FuNction MIN(a,b)
110   IF a<=b THEN
111     RETurn a
112   ELSE
120     RETurn b
121   END IF
130 END DEFine
```

```
100 DEFine PROCedure SCRSV(a$,a)
110   LOCal B$
120   B$="MDV"&a&"_"&a$
130   SBYTES B$,131072,32768
140 END DEFine
150 DEFine PROCedure SCRLD(a$,a)
160   LOCal B$
170   B$="MDV"&a&"_"&a$
180   LBYTES B$,131072
190 END DEFine
```

```
100 DEFine PROCedure update(a$)
110   a$=upper$(a$)
120   IF "MDV"INSTR a$=1 THEN
130     DELETE a$
140     SAVE a$
150   ELSE
160     a$="MDV1 "&a$
170     update a$
180   END IF
190 END DEFine
```

```
100 DEFine PROCedure FILES(a)
110   IPERWINDOW
120   SELect ON a
130     ON a=1:CLS:DIR mdv1_
140     ON a=2:CLS#2:DIR#2;MDV2_
150     ON a=3:CLS#1:CLS#2:DIR mdv1_:DI
R#2;MDV2_
160   END SELect
170 END DEFine
```

```
100 DEFine PROCedure IPERWINDOW
110   OPEN#200,scr_512x256a0x0
120   CLS#200
130   CLOSE#200
140   WINDOW#1,224,192,32,16
150   WINDOW#2,224,192,256,16
160   WINDOW#0,448,42,32,214
170   BORDER#0,1,1
180   CLS#0
190   BORDER#1,1,3
200   CLS#1
210   BORDER#2,1,5
220   CLS#2
230 END DEFine
```

```
100 DEFine PROCedure NORMWINDOW
101   OPEN#200,scr_512x256a0x0
102   CLS#200
103   CLOSE#200
110   WINDOW#0,448,40,32,216
120   WINDOW#1,448,200,32,16
130   WINDOW#2,448,200,32,16
200 END DEFine
```

```
100 DEFine FuNction BIN$(a)
101 LOCal a$
110 a$="";REPeat 1
120 a$=a$&(a-2*INT(a/2));a=INT(a/2):I
F a)0 THEN NEXT 1
130 RETurn a$
140 END DEFine
```

```
100 DEFine FuNction BIN(a$)
101  LOCal p,t
102  t=0
110  FOR p=1 TO LEN(a$)
120   t=t+(2^(LEN(a$)-p))*((a$(p))="1
")
130  NEXT p
140  RETurn t
200 END DEFine
```

```
100 DEFine PROCedure regtime(n);LOCal
 a,b,c,d,e,f
110 INPUT#n;"ANNO)"!a\"MESE)"!b\"GIOR
NO)"!c\"ORE)"!d\"MINUTI)"!e\"SECONDI)
"!f:SDATE a,b,c,d,e,f:time n:END DEFi
ne
```

```
32694 DEFine PROCedure time(n):PRINT#
n;DATE$:END DEFine
```

```
100 DEFine FuNction UPPER$(a$):LOCal
i
110  FOR i=1 TO LEN(a$):IF CODE(a$(i)
)>96 AND CODE(a$(i))<123 THEN a$(i)=C
HR$(CODE(a$(i))-32)
120 RETurn a$
130 END DEFine
```

```
100 DEFine FuNction LOWER$(a$):LOCal
i
110  FOR i=1 TO LEN(a$):IF CODE(a$(i)
)>64 AND CODE(a$(i))<91 THEN a$(i)=CH
R$(CODE(a$(i))+32)
120  RETurn a$
130 END DEFine
```

```
10 DEFine FuNction trunc(n):RETurn n-
INT(n):END DEFine
```

```
32640 DEFine FuNction DECX(a$)
32641 LOCal i,p,Z:p=0
32642 a$=UPPER$(a$):FOR i=1 TO LEN(a$
)
32643 Z=CODE(a$(i)):IF Z>47 AND Z<58
THEN p=p+(Z-48)*16^(LEN(a$)-i):ELSE I
F Z>64 AND Z<71 THEN p=p+(Z-55)*16^(L
EN(a$)-i):ELSE p=0:GO TO 32645
32644 NEXT i
32645 RETurn p
32646 END DEFine
```

```
32647 DEFine FuNction HEX$(n)
32648 LOCal a,b,c,d,r$,Z
32649 a=0:b=0:c=0:d=0:Z=0
32650 IF n>65536*16-1 THEN GO TO 3265
6
32651 IF n>65535 THEN Z=INT(n/65536):
n=n-Z*65536
32652 IF n>4095 THEN a=INT(n/4096):n=
n-a*4096
32653 IF n>255 THEN b=INT(n/256):n=n-
b*256
32654 IF n>15 THEN c=INT(n/16):n=n-c*
16
32655 IF n>0 THEN d=INT(n)
32656 r$=esa$(Z)&esa$(a)&esa$(b)&esa$
(c)&esa$(d):RETurn r$:END DEFine
```

```
32657 DEFine FuNction esa$(n):LOCal r
$
32658 IF n<10 THEN r$=n:ELSE r$=CHR$(
55+n)
32659 RETurn r$:END DEFine
```

```
100 DEFine FuNction fact(n)
110  IF n>1 THEN
120   RETurn n*fact(n-1)
130  ELSE
140   RETurn n
150  END IF
160 END DEFine
```

Superbasic

```
10 REMark indovina in 10 tentativi
20 a=RND(50)
30 FOR all_game=1 TO 10
40 INPUT "Tentativo n."&all_game;"--> 
";tenta
50 IF tenta=a THEN BEEP 9000,0:PRINT 
"OKAY!":EXIT all_game
60 IF tenta>a THEN PRINT "Troppo gran
de":NEXT all_game
70 IF tenta<a THEN PRINT "Troppo picc
olo":NEXT all_game
80 END FOR all_game
90 PRINT "Premi S per un'altra partit
a"
100 IF INKEY$(-1)=="S" THEN GO TO 20
```

```
10 REMark multi_FORMAT
20 prepara_schermo
30 chiedi_nome
40 quante_volte
50 formatta_cartuccia
60 DEFine PROCedure prepara_schermo:M
ODE 8:CLS:END DEFine
70 DEFine PROCedure chiedi_nome
80 INPUT"Come chiamo la cartuccia? "j
nome$
90 IF LEN(nome$)>10 THEN PRINT "TROPP
O LUNGO":GO TO 80
100 PRINT"OK"
110 END DEFine chiedi_nome
120 DEFine PROCedure formatta_cartucc
ia
130 FOR formatta=1 TO volte:FORMAT "m
dv1_"&nome$
140 END DEFine formatta_cartuccia
150 DEFine PROCedure quante_volte
160 INPUT "Quante volte formatto la c
artuccia? "jvolte
170 END DEFine quante_volte
```

```
10 REMark ***************************
20 REMark *                         *
30 REMark *    C A P S  L O C K     *
40 REMark *         per  QL         *
50 REMark *       (c)  1985         *
60 REMark *  by Roberto Previtera    *
70 REMark *                         *
80 REMark ***************************
100 a=RESPR (185):x=0
110 REPeat loop
120 IF EOF THEN EXIT loop
130 READ dati:POKE a+x,dati:x=x+1
140 END REPeat loop
150 DATA 97,0,0,32,50,57,0,2,128,136,
12,65,0,0,102,0,0,10,67,250,0,152,96,
0,0,98,67,250,0,136,96,0,0,90,67,250,
0,116,52,57,0,2,142,28,180,105,0,4,10
3,0,0,70,51,252,0,0,0,2,142,36,50,57,
0,2,142,30,4,65,0,10,51,193,0,2,142,3
0,51,121,0,2,142,28,0,4,51,124,0,10,0
,6,51,121,0,2,142,24,0,8,210,121,0,2,
142,26,51,65,0,10,52,120,0,198,78,146
,75,250,0,64,42,136,78,117,32,122,0,5
6,54,60,255,255,52,60,0,8,112,7,78,67
,112,17,50,60,0,0,54,60,255,255,78,67
,78,117,0,0,0,7,0,0,0,0,0,0,0,0,67,65
,80,83,32,79,78,32,67,65,80,83,32,79,
70,70,0,0,0,0,101
160 SAVE mdv1_capslock,200:SBYTES mdv
1_caps_c,a,x
170 STOP
200 a=RESPR(184):LBYTES mdv1_caps_c,a
:POKE 164002,96:POKE_W 164004,3846:PO
KE_W 167850,20217:POKE_L 167852,a:NEW
```



```
QL MACRO ASSEMBLER VERSION 1.5
  LOC       OBJECT          STMT        SOURCE STATEMENT
                              1 * * * * * * * * * * * * * * * * * * * * * * * * *
                              2 *                                               *
                              3 *            C A P S   L O C K                  *
                              4 *                                               *
                              5 *                 per  QL                       *
                              6 *                                               *
                              7 *               (c)  1 9 8 5                    *
                              8 *                                               *
                              9 *            by  Roberto Previtera              *
                             10 *                                               *
                             11 * * * * * * * * * * * * * * * * * * * * * * * * *
                             12
                             13
                             14 * definizioni di alcune utili costanti
```

```
                                15
=00C6                           16 ut_con   equ     $c6             Vett. crea wind
=0007                           17 io_sstrg equ     $7              D0=Send string
=0011                           18 sd_tab   equ     $11             D0=tab cursor
=00028088                       19 sv_caps  equ     $28088          Caps mode
=00028E1C                       20 sd_xsize equ     $28e1c          Punt. larg. #0
=00028E1E                       21 sd_ysize equ     $28e1e          Punt. alt.  #0
=00028E18                       22 sd_xmin  equ     $28e18          Punt. xpos  #0
=00028E1A                       23 sd_ymin  equ     $28e1a          Punt. ypos  #0
=00028E24                       24 sd_ypos  equ     $28e24          Punt. ypos  #0
                                25
                                28 * * * * * * * * * * * * * * * * * * * * * * * * * *
                                29
                                30 * * * *  I N I Z I O   P R O G R A M M A  * * * *
                                31
                                32 * * * * * * * * * * * * * * * * * * * * * * * * * *
                                33
                                34
0000'  6100 0020                35          bsr     defwind         GO SUB defwind
0004'  3239 0002 8088           36          move.w  sv_caps,d1      Salva sv_caps
000A'  0C41 0000                37          cmpi    #0,d1           Controlla val
000E'  6600 000A                38          bne     capson          IF caps THEN
                                39 *                                GO SUB capson
                                40
                                41 * * * * * * * * *  CAPS OFF  * * * * * * * * *
                                42
0012'  43FA 0098                43 capsoff  lea     spento,a1       String=spento
0016'  6000 0062                44          bra     print           GO TO print
                                45
                                46 * * * * * * * * *  CAPS ON  * * * * * * * * *
                                47
001A'  43FA 0088                48 capson   lea     acceso,a1       String=acceso
001E'  6000 005A                49          bra     print           GO TO print
                                50
                                51 * * CONTROLLA SE LA FINESTRA 0 E' CAMBIATA * *
                                52
0022'  43FA 0074                53 defwind  lea     windef,a1       A1 punta para-
                                54 *                                metri di wind-
                                55 *                                dow
0026'  3439 0002 8E1C           56          move    sd_xsize,d2     Largh. window0
                                57 *                                in d2
002C'  B469 0004                58          cmp.w   4(a1),d2        se  window 0 =
0030'  6700 0046                59          beq     rit             window nuova
                                60 *                                then GO TO rit
0034'  33FC 0000 0002 8E24      61          move.w  #0,sd_ypos      Cursor ypos=0
003C'  3239 0002 8E1E           62          move.w  sd_ysize,d1     Ysize in D1
0042'  0441 000A                63          subi.w  #10,d1          Ysize=Ysize- 1
                                64 *                                riga
0046'  33C1 0002 8E1E           65          move.w  d1,sd_ysize     Modifica wind
004C'  3379 0002 8E1C 0004      66          move.w  sd_xsize,4(a1)  Window nuova =
                                67 *                                window 0
0054'  337C 000A 0006           68          move.w  #10,6(a1)       Ysize di window
                                69 *                                nuova=10(1 riga)
```

```
005A'  3379 0002 8E18 0008    70          move.w  sd_xmin,8(a1)   xpos nuova wind-
                              71 *                                ow=xpos window 0
0062'  D279 0002 8E1A         72          add.w   sd_ymin,d1      Ypos  di window
0068'  3341 000A              73          move.w  d1,$a(a1)       nuva= ypos wind-
                              74 *                                ow 0+ysize wind-
                              75 *                                ow 0
                              76
                              77 * * * * * * * *  CREA LA FINESTRA  * * * * * * * *
                              78
006C'  3478 00C6              79          move.w  ut_con,a2       A2 punta ut_con
0070'  4E92                   80          jsr     (a2)            GO SUB ut_con
0072'  4BFA 0040              81          lea     channel,a5      Salva channel ID
0076'  2A88                   82          move.l  a0,(a5)
0078'  4E75                   83 rit      rts                     RETurn
                              84
                              85 * * INVIA UNA STRINGA DI CARATTERI A UN CANALE *
                              86
007A'  207A 0038              87 print    move.l  channel,a0      A0=channel ID
007E'  363C FFFF              88          move.w  #-1,d3          D3=timeout
0082'  343C 0008              89          move.w  #8,d2           D2=numero char.
0086'  7007                   90          moveq   #io_sstrg,d0    D0=Send string
0088'  4E43                   91          trap    #3              GO SUB io_sstrg
008A'  7011                   92          moveq   #sd_tab,d0      D0=tab cursor
008C'  323C 0000              93          move.w  #0,d1           cursor at 0
0090'  363C FFFF              94          move.w  #-1,d3          D3=timeout
0094'  4E43                   95          trap    #3              GO SUB sd_tab
                              96
                              97 * * * * * * * *  FINE PROGRAMMA  * * * * * * * * *
                              98
0096'  4E75                   99 fine     rts                     RETurn;END
                             100
                             101 * * * * *  PARAMETRI NUOVA WINDOW  * * * * * * *
                             102
0098'  0000 0007             103 windef   dc.b    0,0,0,7         Window nuova
                             104 *                                parametri
                             105 *                                Bordo,colore
                             106 *                                ink,paper
009C' =0008                  107          ds.w    4               xsize,ysize
                             108 *                                xpos,ypos
                             109
                             110 * * * * * * * *  STRINGA 'CAPS ON' * * * * * * * *
                             111
00A4'  4341 5053 204F 4E20   112 acceso   dc.b    'CAPS ON '      String=capson
                             113
                             114 * * * * * * * * STRINGA 'CAPS OFF' * * * * * * * *
                             115
00AC'  4341 5053 204F 4646   116 spento   dc.b    'CAPS OFF'      String=capsoff
                             117
                             118 * * * *  MEMORIA IDENTIFICATORE  DEL CANALE  * *
                             119
00B4' =0004                  120 channel  ds.l    1               Channel storage
                             121
                             122          end
```

I simulatori di volo

di Luigi Callegari

Considereremo tre programmi di simulazione di volo aereo per lo Spectrum (48K), scelti come i più rappresentativi e famosi: Flight Simulation, Flighter Pilot, Delta Wing. Tutti e tre hanno delle caratteristiche tali da poter essere definiti *simulatori*, e non semplici giochini; caratterizzante in questo senso è la visione esterna con grafica tridimensionale in tempo reale.

Flight Simulation (Psion). È stato uno dei primi programmi per lo Spectrum, nel lontano 1982, ma rimane egualmente assai valido, è un irrinunciabile termine di confronto per tutti gli altri programmi del genere. Il velivolo simulato è un piccolo bimotore.

I comandi disponibili sono: cloche (4 direzioni), timone, flap, carrello d'atterraggio, throttle (gas motori), mappa e stazione radar di riferimento. Come avviene anche negli altri due programmi presentati, è possibile avere sul video una vera e propria cartina indicante con simboli: la nostra posizione, la posizione dei fari radar, la posizione degli aereoporti e delle particolarità geografiche (montagne e laghi) eventualmente visibili.

Il volo è guidato da una serie di basi radar a terra, sulle quali possiamo sintonizzare i nostri radar di bordo per un tranquillo volo strumentale. Sul video, oltre alla visione dello spazio esterno, che occupa i due terzi superiori, abbiamo i seguenti strumenti: tachimetro (in nodi), altimetro (in piedi), ROC (angolo di cabrata), potenza dei motori, carburante, angolo flaps, ILS (sistema atterraggio strumentale guidato da terra) e altri tre strumenti, siglati BCN, RGE, BRG, che formano un evoluto sistema radar per la navigazione.

Il programma, essendo stato sviluppato dai programmatori Psion in collaborazione con i piloti di linea, è molto fedele, anche se a chi non ha mai guidato un bimotore potrebbe sembrare assai lento.

Il difetto maggiore è l'impossibilità di usare i tradizionali joystick, caratteristica presente in tutti gli altri simulatori a noi noti.
Mancano effetti sonori fantasiosi, ma la grafica è ben rifinita, specie per quanto riguarda la strumentazione di volo.

Fighter Pilot (Digital Integration). Questo programma è stato segnalato dalla stampa inglese specializzata come uno dei 10 migliori programmi per Spectrum dell'anno 1983. Sviluppato da un pilota-ingeniere-programmatore, è basato sui dati tecnici dell'F.15 "Eagle", un caccia supersonico della USAF.

La prima nota di differenza, ovvero di superiorità rispetto al precedente, è il grande numero di opzioni iniziali. A parte i 4 possibili livelli di difficoltà (da apprendista a pilota professionista) e la possibilità di scegliere tra controlli da tastiera o con joystick vari, è possibile anche scegliere tra i seguenti modi di funzionamento: pratica di atterraggio, allenamento al volo, atterraggio cieco (niente visione esterna), pratica di combattimento aereo, combattimento effettivo.

I controlli disponibili sono: cloche (4 direzioni), timone, flaps, throttle, carrello, mappa, stazione radar, attivazione del computer da combattimento o di quello d'atterraggio, cannoncini e sospensione

(segue a pag. 62)

Un tasto per volta

H

NORMALE

Istruzione GO SUB. Sposta l'esecuzione del programma alla "subroutine" *xxx*, cioè alla linea numero *xxx*, dato come argomento.

Una subroutine è una sezione isolabile del programma, che dovendo essere eseguita più volte viene scorporata; il programma principale può saltare alla subroutine da qualsiasi punto, e ritornare a questo al termine. Il vantaggio è il risparmio di memoria e, per i puristi, l'eleganza di programmazione (non si scrivono più volte le stesse istruzioni); per contro si ha una leggera perdita di velocità: dovendo ottimizzare i tempi (senza avere problemi di memoria), conviene evitare i salti e programmare linearmente.

Di GO SUB abbiamo già parlato a proposito del suo completamento RETURN (cfr. SC n. 12 pag. 53); per quanto riguarda l'argomento, vale tutto quanto è stato scritto per GO TO nello scorso numero.

Attenzione a non dimenticare il RETURN: l'errore non viene segnalato e il salto resta memorizzato nella catasta dei GO SUB.

SYMBOL SHIFT

Elevamento a potenza. Istruzione molto... potente, a patto che non si richiedano grandissime precisioni e velocità. È in grado, in teoria, di elevare qualsiasi base a qualsiasi esponente, entrambi interi o decimali, in un range utile che ha per estremi (del risultato) 2. 9387359E-39 e 1. 7014118E + 38.

L'elevamento a potenza serve anche, come molti ricorderanno, per l'estrazione di radici il cui indice sia diverso da 2, anche decimale (utile perciò per risolvere equazioni logaritmiche): per esempio, porre come esponente 1/3 significa estrarre la radice cubica.

Nell'uso ricordate che la priorità più alta di tutti gli operatori aritmetici e logici: nei casi dubbi, usate le parentesi.

Attenzione anche ai segni: con una base negativa va in errore; per esempio, elevando -2 al quadrato ottenete -4, a causa della priorità: usando le parentesi si ha errore A *(Invalid argument)*.

Perciò, se l'operando di sinistra può diventare negativo, bisogna memorizzare il segno (con SGN), usare il valore assoluto (ABS) e verificare se l'esponente sia pari o dispari per stabilire il segno del risultato.

Il simbolo usato sullo Spectrum è la freccia rivolta verso l'alto; nei listati appare solitamente un accento circonflesso (solo la punta della freccia): non è un difetto della stampante, la quasi totalità dei computer usa questo simbolo.

Annotiamo infine che quando si tratta di calcolare il quadrato o il cubo di un numero piccolo è decisamente preferibile, per motivi di velocità, eseguire una più banale moltiplicazione.

CAPS SHIFT

"H" maiuscola.

MODO "E"

Estrazione di radice quadrata (SQR *n*). Vi restituisce la radice quadrata dell'argomento, che può essere un numero o una variabile. Un argomento negativo dà ovviamente errore *(Invalid argument)*.

SYMBOL S. IN "E"

Funzione CIRCLE. Questo, Ferrini docet, "lo dice la parola stessa": disegna cerchi (per la cronaca, in senso antiorario). Dovete fornirgli tre argomenti, *x, y,'r:* i primi due indicano le coordinate assolute del centro sul display-file, indipendentemente da dove si trovasse prima la penna virtuale dello schermo; il terzo rappresenta il raggio, misurato in "display-unit" (cioè in pixel). Se il raggio manda la circonferenza fuori dalla finestra dello schermo, si ha una situazione di errore *(Integer out of range)*.

Un'informazione che a volte può tornare utile: al rientro dall'esecuzione, la penna virtuale resta posizionata sulla circonferenza sempre nella stessa posizione relativa, nel quadrante destro inferiore (x + /y-, analiticamente parlando), qualche pixel più in basso rispetto al diametro orizzontale.

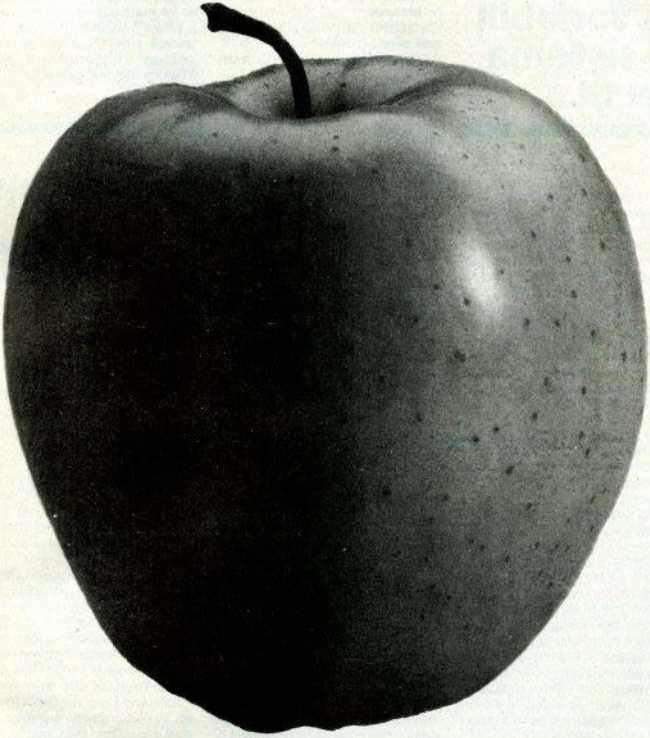
J

NORMALE

Istruzione LOAD, cioè "carica": da nastro, microdrive, o altro. Questa è la primissima istruzione con cui solitamente si incontra l'utente, anche (anzi, soprattutto) quello meno programmatore, che utilizza lo Spectrum solo con cassette già pronte. Ne abbiamo parlato più volte; la sintassi è la stessa di SAVE (cfr. n. 13 pag. 54).

(segue a pag. 63)

Caro piano media, prendi le vitamine.



Secondamano, il media polivitaminico più indicato in Lombardia. (Vitamina B) 70mila copie bisettimanali - (Vitamina C) 700mila lettori, con aumento costante del 30% annuo - (Vitamine A, E, D) Network di 29 testate a inserzioni gratuite come copertura nazionale. Posologia e indicazioni: Usare spesso. L'azione tonificante delle 70mila copie e dei 700mila lettori è stata sperimentata con successo anche sui piani media più abulici. Controindicazioni: Nessuna.

secondamano

Le variabili di sistema del QL

N.B.: W significa che la variabile è a 16 bit, L che è a 32 bit e B che è a 8 bit. Pertanto, le PEEK e le POKE riferite alle variabili di sistema dovranno avere il giusto suffisso: dove c'è B usate PEEK e POKE normali; dove c'è W usate PEEK W e POKE W; dove c'è L usate PEEK L e POKE L.

W $28000 identificazione

L $28004 puntatore SV. HEAP

L $28008 puntatore al primo spazio libero nell'area heap

L $2800C puntatore SV. FREE

L $28010 puntatore SV. BASIC (puntatore allo stack del basic)

L $28014 puntatore SV. TRNSP

L $28018 puntatore al primo spazio libero nell'area dei programmi transienti

L $2801C puntatore SV. RESPR (equivalente al RESPR del superbasic)

L $28020 puntatore SV. RAMT (ultimo byte RAM + 1)

W $2802E numero causale (incrementato ogni 72 microsecondi): analogo alla variabile di sistema FRAMES dello Spectrum

W $28030 contatore degli interrupts non eseguiti

B $28032 se è diverso da 0, state usando un tv; se è uguale a 0, un monitor. Si basa sul tasto premuto all'accensione: F1 o F2

B $28033 È uguale a 0 se lo schermo è attivo, e diverso da 0 se non lo è

B $28034 valore attuale del registro di stato MC

B $28035 valore attuale del registro interrupt PC

B $28037 numero stazione network. Si modifica con il comando NET x, dove x è il numero di stazione desiderato

L $28038 puntatore alla lista dei drivers dell'interrupt livello 2

L $2803C puntatore alla lista del polled tasks

L $28040 puntatore alla lista del task dello scheduler

L $28044 puntatore alla lista dei device drivers

L $28048 puntatore alla lista dei directory device drivers

L $2804C puntatore alla queue di tastiera

L $28050 puntatore alla tavola dei vettori e dei traps

(1 - continua)

Glossario

QUEUE: un buffer usato per l'input/output, lungo generalmente 128 bytes; la struttura di una queue verrà esaminata in uno dei prossimi articoli.

HEAP: catasta; uno spazio usato dai programmi L/M per dati e stack.

TRAPS: istruzioni L/M equivalente alla RST dello Z80 (ce ne sono 16); equivale a un interrupt, e chiama la routine puntata dall'apposito vettore.

VETTORE: puntatore a una routine, può essere a 8, 16 o 32 bit; nel QL quelli su ROM sono tutti a 16 bit.

TAVOLA: area di memoria adibita a immagazzinare dati o puntatori dello stesso tipo; per esempio, la tavola dei vettori è l'area di memoria che contiene tutti i vettori.

DEVICE DRIVER: routine incaricata di gestire le comunicazioni con una periferica; sono periferiche lo schermo, i microdrives, le interfacce seriali...

DIRECTORY DEVICE DRIVER: routine incaricata di gestire la comunicazione con periferiche di tipo complesso per es. con struttura a files: microdrives, Winchester, floppy che richiedono l'uso e la gestione di una o più directories.

TASK SCHEDULER: Lavoro che la CPU deve eseguire. Routine del QDOS, gestita da interrupt, che spartisce il tempo della CPU fra i jobs attivi; è l'organo di gestione del multitasking.

POLLING INTERRUPTS: routines eseguite 50 volte al secondo; per esempio scansione tastiera, aggiornamento variabili di sistema; refresh memoria video... nel QL vi sono tre livelli (priorità) di interrupt: il 2, il 5 e il 7; il QDOS usa solo il 2 (per tastiera e multitasking); gli altri livelli sono a disposizione; il livello 7 è l'unico non mascherabile.

Assembly

di Roberto Previtera

L'assembler 68008

Questo mese cominciamo a parlare di linguaggio macchina anche nell'ambito del QL, che come ormai molti sapranno è dotato del microprocessore 68008, molte potente ma anche più complesso dei comuni microprocessori a 8 bit.

Per prima cosa analizziamo quindi brevemente la struttura del microprocessore, dalla quale è già possibile capire le grandi possibilità che vengono offerte al programmatore.

La struttura *interna* della CPU è a 16/32 bit, cioè le istruzioni sono sempre lunghe 16 bit (word) e devono sempre cominciare a indirizzi pari, mentre i registri sono a 32 bit (long word), fatta eccezione per il Program Counter, a 20 bit, e per lo Status Register, a 16 bit.

Esistono 8 registri DATA (D0-D7), sui quali è possibile eseguire qualsiasi operazione nei formati 8-16-32 bit; sono inoltre disponibili per il programmatore 8 registri ADDRESS (A0-A7), sui quali, sebbene non siano implementate tutte le operazioni, sono disponibili alcune interessanti opzioni, come il *post-incremento* e il *predecremento* o l'indirizzamento indicizzato.

Come già accennato prima, oltre a questi registri ci sono anche il *Program Counter*, capace di indirizzare direttamente 1 megabyte, e lo *Status Register* di 16 bit, che può dirsi diviso in due parti: il byte meno significativo contiene tutti i flag attivati da operazioni condotte su uno qualsiasi degli altri registri, mentre il secondo byte (che come vedremo in seguito non è sempre modificabile) contiene:

La maschera degli interrupts, costituita dai primi tre bit, dei quali però il secondo viene ignorato nel 68008, che può quindi codificare fino a tre livelli di interrupts (quest'argomento verrà chiarito meglio in seguito).

Il flag di modo: indica in che modo il microprocessore sta lavorando, dato che nel 68008 sono disponibili due modi di lavoro, per non causare troppa confusione sappiate per ora che questa distinzione è molto utile affinchè i sistemi operativi non interferiscano con i programmi.

Il flag di modo TRACE: quando questo flag è settato il microprocessore esegue un interrupt ogni volta che è stata eseguita una singola istruzione.

Passiamo ora a descrivere, anche se solo superficialmente, le istruzioni che possono generare più confusione esaminando un listato assembler 68008.

L'istruzione più potente e versatile è senza dubbio la MOVE, equivalente alla LD o alla ST di altri microprocessori: ha il compito di spostare dati dei tre differenti formati.

L'istruzione TRAP genera un interrupt software, il cui vettore può variare tra 16 diversi; il suo utilizzo è indispensabile per chiamare le routines del QDOS, accessibili con le prime quattro TRAP; la TRAP provoca anche il passaggio da modo *utente* al modo *supervisore*, che altrimenti sarebbe impossibile eseguire, dato che alla chiamata di un qualsiasi programma caricato dall'esterno il microprocessore si trova in modo utente, e che da questo stato non è possibile modificare lo SR, pena un crash di sistema.

Un esempio: CAPS LOCK monitor

Dopo aver dato un'occhiata al microprocessore, analizziamo il disassemblato della routine che vi presentiamo questo mese: si tratta di un semplice programma che visualizza lo stato di CAPS LOCK direttamente sullo schermo, essendo il QL sprovvisto di un qualsiasi altro indicatore.

La routine non fa uso del multitasking, che verrà analizzato in una prossima puntata, ma è pilotata direttamente dal sistema operativo tramite la variabile SV CSUB (1 *long word*), che in effetti non deve contenere un puntatore, ma la routine stessa, che dovrà essere lunga solo quattro byte, appena sufficienti per un salto quindi risolto caricando dall'indirizzo 167850 un altro salto, questa volta assoluto, in grado di puntare alla routine caricata nella parte alta della memoria.

Il programma una volta chiamato controlla le dimensioni della finestra 0, le sottrae una riga e apre una finestra esattamente sotto di essa; viene poi effettuato un controllo sulla variabile SV. CAPS e in base al risultato ottenuto viene chiamata la routine di sistema IO.SSTRG per stampare il messaggio e la SD.TAB per riposizionare il cursore.

Per ottenere una copia funzionante di CAPS LOCK, dovete digitare il listato BASIC, salvarlo su Microdrive (per non perdere i dati in caso di errori nell'inserimento dei codici), e dare semplicemente il RUN.

Il programma provvederà a salvare su Microdrive il ling. macchina e una riga BASIC adibita al caricamento dei due salti relativi; subito dopo questa riga si cancellerà; da quel momento la routine sarà attiva.

collaborazioni

Chi desidera collaborare alla nuova sezione SPAZIO QL può farlo nella stessa forma prevista per lo Spectrum, inviando testi e programmi alla redazione. Si raccomanda di scrivere a macchina o con word processor.

I programmi dovranno essere listati e registrati su cartridge, che verrà sostituita a stretto giro di posta con una cartuccia vuota; ciò non impegna in nessun modo la redazione a pubblicare il materiale inviato.

Il *QL User Club* di Milano, che cura lo SPAZIO QL su Sinclair Computer, è disponibile per contatti e scambi di informazioni con tutti gli utenti QL. Il recapito è:

QL User Club Milano
c/o Marco Mussini
via Tajani 9
20133 Milano MI

Super basic

a cura di M. Mussini
QL User Club Milano

Prenderemo in esame le singole istruzioni che il Superbasic ci offre cercando di analizzarne a fondo modo d'uso, pregi e difetti, e di fare ogni volta un parallelo con la corrispondente istruzione del basic convenzionale.

FOR

In Superbasic se ne hanno due tipi, funzionalmente equivalenti ma formalmente differenti; essi sono la forma *breve* e la forma *lunga*.

Un loop FOR...NEXT in forma breve è molto simile all'equivalente basic standard; la sintassi fondamentale è

FOR *variabile* = *valore iniziale* TO *valore finale* STEP *incremento: [istruzioni da ripetere];* NEXT *variabile*

Esempi:

(1) FOR pqr = 1 TO 100 STEP 4.5: PRINT pqr: NEXT pqr

(2) FOR scrivi = 29.2 TO 66: PRINT "n = "; scrivi: NEXT scrivi

(3) FOR c = 1E6 TO 1E7 STEP 1E2: print c: NEXT c

(4) FOR breve = 100 TO 200: print breve

(5) FOR out = 1 TO 10: FOR in = out TO 20 STEP 3: PRINT out, in: NEXT in: NEXT out

(6) FOR super-loop = 1 TO 10, 3 TO 15, A% TO A% + 8, - 2 TO 4.9 STEP 5, 100 TO 1000: PRINT superloop

Notate che:

1 - la variabile di un ciclo FOR può solo essere floating point; ciò spiega in parte la relativa lentezza dei FOR...NEXT del Superbasic: sul QL le variabili floating point hanno un range immenso!;

2 - l'incremento può essere omesso. in questo caso, il computer assume per default il valore 1;

3 - alcune spaziature non sono necessarie;

4 - la variabile dopo il NEXT non può essere omessa (come nel basic Microsoft);

5 - le istruzioni da ripetere possono essere più di una;

6 - il nome della variabile FOR può essere lungo fino a 255 fra lettere, cifre e separatori ( ), a differenza dello Spectrum, che permette al massimo un solo carattere alfabetico; ciò permette di avere più di 26 cicli FOR...NEXT annidati e di rendere autoesplicativo il nome della variabile di controllo;

7 - i parametri del ciclo FOR...NEXT possono essere espressi in forma esponenziale, o anche essere delle variabili (in questo caso anche intere o stringa: vengono convertite automaticamente nel giusto formato);

8 - forse la maggior comodità del FOR breve, il NEXT può essere omesso (v. esempio 4); se si omette il NEXT, non è evidentemente possibile annidare i cicli brevi (esempio 5);

9 - infine, la più importante innovazione: la possibilità di definire diversi intervalli di valori per la variabile di controllo (esempio 6): si può anche specificare uno STEP diverso per ogni intervallo. Questa possibilità ne rende estremamente flessibile l'uso.

Vediamo ora le peculiarità del FOR..NEXT lungo. Un esempio di come funziona è il listato 1.

(list a pag. 40)

Il NEXT viene soppiantato dalla nuova END FOR (usata per definire la fine di un ciclo); ora NEXT svolge la funzione di ripetere il ciclo prima che siano state eseguite tutte le istruzioni al suo interno: insomma un END FOR parziale, eseguito qualora siano state raggiunte certe condizioni, che richiedano di ripetere immediatamente il ciclo (linee 60 e 70); se questo NEXT dovesse provocare la fine del ciclo (per aver raggiunto il limite di 10 tentativi) l'esecuzione del programma proseguirebbe con la prima linea incontrata appena usciti dal ciclo (cioè la 90) e non, come sullo Spectrum, con quella immediatamente seguente il NEXT stesso.

Naturalmente è possibile fare a meno di END FOR e usare il NEXT alla vecchia maniera, anche se è decisamente consigliabile fare uso del nuovo sistema che il Superbasic mette a disposizione.

Tutte le osservazioni fatte a proposito del FOR *breve* rimangono valide, fatta eccezione logicamente per le caratteristiche che lo distinguono dalla forma lunga; è inutile sperare di poter omettere il NEXT in una struttura così complessa.

Notate infine l'istruzione EXIT in riga 50, che permette di uscire dal loop senza creare problemi di sorta con lo stack; quando viene eseguita, il suo effetto è quello di far continuare l'esecuzione con la prima istruzione al di fuori dal ciclo (come un GOTO 90).

DEF PROC

Rappresenta, insieme con la gemella DEF FN, l'innovazione più appariscente.

Le due istruzioni sono fondamentalmente simili e le esaminiamo una di seguito all'altra.

(list a pag. 40)

Il programma del listato 2, una comoda utility per formattare a ripetizione le cartucce più "taccagne", contiene alcuni semplici esempi di ciò che si può fare con la DEF PROC. Avrete certamente notato che la sua impostazione generale somiglia molto a una più familiare, e cioè la GOSUB-RETURN, cui corrispondono DEF PROC e END DEF. La nuova struttura presenta vantaggiose differenze:

1. Modificare i numeri di linea della subroutine non compromette il buon funzionamento del programma, perchè per chiamarla non si usa più un numero di linea, ma una specie di LABEL (etichetta), il cui valore viene automaticamente ag-

giornato dal computer ogni volta che si dà RUN o che si modifica il listato: questa operazione viene segnalata dal messaggio "PROC/FN cleared".

2. Il nome della subroutine può avere una lunghezza qualsiasi, (fino a 255 caratteri), quindi sufficiente a scrivere intere frasi. Ne discende la possibilità di rendere chiari e autoesplicativi i nomi delle routines, in modo da capire a che servano, sia quando se ne trova la chiamata sia quando se ne trova la definizione. Non c'è paragone, quindi, con i nebulosi numeri di linea del basic convenzionale.

3. La leggibilità e la chiarezza interpretativa dei listati migliorano molto, rendendo sempre meno necessario l'uso di REMark.

4. Se una subroutine ha bisogno di alcuni parametri per funzionare, è molto più facile capire di che parametri si tratta e soprattutto che si riferiscono alla subroutine: confrontate questi due listati:

```
10 LET XXX=1.25E23: LET
A$="PIPPO": LET WED=2+T*2:
GOSUB 3247
```

```
10 SCRIVLVALORI 1.25E23, "PIP-
PO", 2+T*2, altra_variabile
```

A parte l'evidente compattezza del secondo, si noti anche che è immediato il collegamento logico di quei valori alla chiamata della subroutine; si può forse dire lo stesso del primo caso, ma non vedendovi assegnare alcun valore ad "altra variabile", si può pensare che essa non rientri nella lista paramentri della subroutine; nel secondo caso sparisce anche questa incertezza si vedono subito tutti i valori coinvolti.

5. Un'altra fondamentale differenza si cela in questi due listati, apparentemente equivalenti:

```
10 FOR I=1 TO 10:GOSUB
20:NEXT I
20 PRINT "I=";I:RETURN
```

```
10 FOR I=1 TO 10:WRITE I
20 DEF PROC WRITE
(NUMERO):PRINT''I='';
NUMERO:END DEF
```

Il primo fornisce questo output:

```
I=1
I=2
I=3
...
I=10
I=11

RETURN WITHOUT GOSUB
```

Infatti, terminato il ciclo principale, il computer esegue la linea 20 come una linea normale, visualizzando perciò "I=11" e, incontrando il RETURN, emette il messaggio d'errore.

In Superbasic ciò non avviene: quando l'interprete incontra delle DEF PROC, non cerca di interpretarle, ma le salta direttamente, esattamente come con DEF FN. Anche questo aiuta a ridurre i bug dei programmi basic.

6. Per potente che sia, la DEF PROC resta pur sempre una stretta parente del GOSUB, e quindi anch'essa necessita di uno stack appositamente riservato, allo scopo di «ricordarsi» da dove la subroutine è stata attivata, in modo che, a esecuzione avvenuta, si possa tornare all'istruzione immediatamente successiva alla chiamata. Come ogni altro stack anche questo non ha dimensioni illimitate e quindi se lo si riempie in continuazione rischia di superare i «limiti di sicurezza», impedendo l'esecuzione di altri GOSUB e bloccando il computer.

Si provi a digitare questa riga su uno Spectrum e su un QL:

```
10 GOSUB 10
```

Dopo 25 secondi di esecuzione nello Spectrum 48K, e dopo 50 secondi nel QL inespanso, il computer si arresta emettendo il messaggio di errore OUT OF MEMORY, dovuto al saturamento dello stack; dopodichè lo Spectrum rifiuta qualunque comando, eccetto NEW, mentre il QL torna subito e automaticamente alla normalità, ripulendo lo stack. Problemi dello stesso tipo sorgono anche se si fa un uso sconsiderato della ricorsività: per esempio una riga come

```
10 DEF PROC VAI:VAI:END DEF
```

(impostare VAI -ENTER- per far partire il programma)

dà luogo alle stesse reazioni. Se però si fa in modo che il numero delle iterazioni sia limitato, non appena queste saranno terminate il computer il computer eliminerà automaticamente lo stack dalla memoria.

Ciò permette di scrivere delle subroutine ricorsive, che si autorichiamano (non all'infinito!), senza i rischi cui si andrebbe incontro usando GOSUB.

Nell'ambito della DEF PROC ci sarebbe ancora da fare un discorso sulle variabili locali; ne parleremo a proposito di DEF FN.

Come vedete, DEF PROC è decisamente vantaggiosa rispetto al suo "antenato" GOSUB, e costituisce uno dei punti di forza del Superbasic per la programmazione strutturata. Non c'è da stupirsi, quindi, se la Sinclair stessa ha spietatamente definito "ridondante" il GOSUB, implementandolo per completezza: è realmente possibile farne a meno.

DEFine FuNction

Consiste in una versione molto estesa della vecchia omonima cui siamo abituati, usando altri tipi di basic.

Il principale svantaggio della normale DEF FN era la scarsa versatilità: la definizione poteva al massimo contenere espressioni stringa o numeriche più o meno lunghe; per esempio:

```
10 DEF FN A(R)=R**3.14
```

che ritorna l'area di un cerchio di raggio R.

Questo tipo di DEF FN può bastare in molti casi, ma talora si sente il bisogno di inserirvi dei comandi, ad esempio dei FOR...NEXT, o magari di rendere ricorsive le nostre funzioni, per esempio per implementare la funzione di fattoriale:

```
10 DEFine FuNction fatt (numero)
20 IF numero=1 then RETurn nu-
mero:ELSE RETurn numero * fatt
(numero-1)
30 END DEF fatt
```

La linea 20 contiene tutte le novi-

SPAZIO QL

tà: innanzitutto viene effettuato un controllo sul parametro ricevuto: se è 1, viene ritornato il numero stesso; altrimenti viene prima moltiplicato per se stesso diminuito di 1. Questo processo continua a decrementare la variabile, finchè il suo valore diviene pari a 1; in questo caso il ciclo ricomincia in senso inverso, per il «trasporto» dei risultati intermedi fino alla prima subroutine chiamante; infine viene ritornato il risultato complessivo al basic. Tutto ciò sarebbe impossibile con la DEF FN dello Spectrum: digitando queste righe

```
10 DEF FN a() = FN a()

PRINT FN a()
```

si manda il computer in tilt: dopo alcune iterazioni, il computer stampa «PARAMETER ERROR» e se si chiede il listato si noterà che è stato danneggiato da strani simboli lampeggianti; dopo alcune pressioni di tasto, il computer s'inceppa.

Sul QL tutto ciò non avviene, eccetto, ovviamente, l'inevitabile saturazione dello stack.

Per quanto riguarda DEF FN, non c'è molto altro da dire, in quanto essa è in tutto e per tutto simile a DEF PROC. La sostanziale differenza è che una FuNzione deve poter passare al programma chiamante un risultato, e questo è possibile mediante l'uso del comando RETURN *nome variabile*; ecco un esempio:

```
10 DEF FN MAX (A,B)
20 IF A>B THEN RETURN A
30 IF B>A THEN RETURN B
40 IF A=B THEN RETURN A
50 END DEF
```

Questa funzione ritorna il maggiore fra due numeri; se i due numeri sono uguali (linea 40) ne ritorna uno (a piacere vostro). RETURN può essere seguita anche da espressioni piú complesse, per esempio

```
10 DEF FN INSULTI(NOME$): RETURN NOME$ & ",SEI UN ASINO!":END DEF
20 INPUT "COME TI CHIAMI?"; A$:PRINT INSULTI$(A$):GOTO 20
```

Notate che la variabile usata nella chiamata può non avere lo stesso nome di quella impiegata dalla funzione; in pratica, i parametri passati alla routine acquistano nel suo ambito un altro nome.

Notate soprattutto che si vuole che la funzione ritorni un parametro stringa, il suo nome deve terminare con un "$", mentre se si desidera che il parametro di ritorno sia numerico il dollaro non ci deve ovviamente essere.

Se questa regola non viene rispettata, il computer tenta di convertire il valore desunto dalla RETurn nel formato indicato da questo suffisso, e se non ci riesce si blocca con il messaggio "ERROR in EXPRESSION": non è facile convertire "PIPPO" in formato floating point!

Per concludere l'argomento DEF PROC e DEF FN ci resta da parlare delle variabili locali.

```
10 FOR I = 1 TO 3:OUTPUT
20 DEF PROC OUTPUT
30 FOR I = 1 TO 5:PRINT "CIAO"
40 END DEF
```

A prima vista si può pensare che dando RUN si vedrà apparire 15 volte la scritta "CIAO"; in realtà le cose non stanno cosí: apparirà per 5 volte soltanto. Questa irregolarità è dovuta al fatto che la stessa variabile viene usata sia dal programma principale sia dalla subroutine.

Ecco quello che succede passo passo:

-linea 10: viene preparato un loop da 1 a 3; I = 1

-linea 10: viene richiamata la subroutine OUTPUT.

-linea 20: il computer localizza la subroutine.

-linea 30: viene preparato un loop da 1 a 5, controllato dalla variabile I.

-linea 30: I = 1

-linea 30: stampa della scritta "CIAO"; I = 2

-linea 30: stampa della scritta "CIAO"; I = 3

-linea 30: stampa della scritta "CIAO"; I = 4

-linea 30: stampa della scritta "CIAO"; I = 5

-linea 30: stampa della scritta "CIAO"; fine del ciclo

-linea 40: la routine passa il controllo al programma principale

-linea 10: I è maggiore del limite stabilito (3); fine del ciclo.

Con un programma semplice come quello sopra riportato la correzione è semplice (o meglio, l'errore non dovrebbe venire commesso), ma quando i programmi hanno una certa mole può risultare difficoltoso controllare i nomi delle variabili usate in tutti i punti del programma; d'altronde può essere necessario che la routine interagisca con le variabili del programma principale. Per rendere possibili entrambe le cose, il Superbasic è dotato del comando

LOCAL *nome variab1, (nome variab2, nome variab3,...)*

che fa sí che le variabili specificate acquistino un valore diverso da quello che hanno normalmente: un po' come i parametri delle FuNzioni. Insomma, se inseriamo la linea

```
25 LOCal I
```

il programma gira regolarmente: il computer considera le due variabili I come se avessero nomi distinti, eliminando quindi ogni possibile interferenza.

Concludendo, rammentiamo che in virtú della possibilità (apparente) di estendere il numero delle parole chiave del QL mediante DEF PROC e DEF FN, sulle pagine di "spazio QL" verrà pubblicato a puntate un gigantesco agglomerato di nuove PROCedure e FuNzioni che presto vi metterà a disposizione quasi 60 nuove keywords.

QL END

SEIKOSHA

NON AVRAI ALTRA STAMPANTE

Seikosha ti invita nel meraviglioso mondo delle sue stampanti.
Un mondo fatto di progresso, di elevatissima qualità, velocità e silenziosità di stampa.
Seikosha oggi ti propone la più vasta gamma di stampanti nate per esaltare le prestazioni di ogni tipo di computer.
All'altezza di ogni esigenza, anche della tua che usi i Personal Computer Sinclair.
Piccola e compatta, dalle prestazioni generose, GP 50 S con 35 caratteri al secondo e 32 colonne, è la stampante ideale per risolvere con soddisfazione le prime esigenze di stampa di chi usa lo ZX Spectrum.
Se possiedi anche l'interfaccia 1, niente di meglio della stampante GP 500 S con 50 caratteri al secondo e 80 colonne che ti consente utilizzi anche di tipo gestionale.
Se lavori con un computer Sinclair QL, non puoi rinunciare agli 80 caratteri al secondo e 80 colonne anche Near Letter Quality a 20 caratteri per secondo del modello SP 800 IQL.
Se poi le tue esigenze sono altamente professionali, la stampante BP 5420 A con 136 colonne, 420 caratteri al secondo, anche Near Letter Quality a 104 caratteri al secondo, rende ancor più grande il tuo Sinclair QL.
Seikosha e Sinclair: una coppia che và d'amore e d'accordo.

SEIKOSHA
Distribuzione esclusiva: GBC Divisione R...

I 25 games più votati dai nostri lettori:

1. Knight Lore	(Ultimate)
2. Sabre Wulf	(Ultimate)
3. Match Point	(Psion)
4. Chequered Flag	(Psion)
5. D.T. Decathlon	(Ocean)
6. Mugsy	(Melbourne)
7. Ant Attack	(Quicksilva)
8. Underwurlde	(Ultimate)
9. Jet Set Willy	(Soft. Projects)
10. Jet Pac	(Ultimate)
11. Atic Atac	(Ultimate)
12. Match Day	(Ocean)
13. Manic Miner	(Soft Projects)
14. Ghostbuster	(Activision)
15. Pole Position	(Atari)
16. World Cup Football	(Artic)
17. Fighter Pilot	(Digital Integr.)
18. Hulk	(Adventure Int.)
19. Psytron	(Beyond)
20. Hobbit	(Melbourne)
21. Lords of Midnight	(Beyond)
22. Stop the Express	(Psion/Hudson)
23. Flight Simulation	(Psion)
24. Maziacs	(Dk' Tronics)
25. Sports Hero	(Activision)

sinclair *parade*

Un po' di Pascal

a cura di Monica Fumagalli

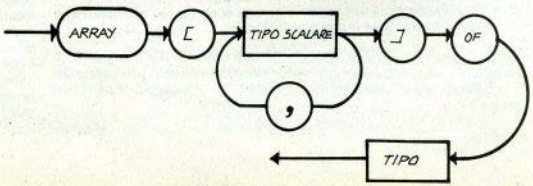
I tipi strutturati possono essere definiti attraverso aggregazioni di tipi semplici; i meccanismi di strutturazione sono quattro: array, record, set e file.

Array

Un array rappresenta una collezione ordinata di un numero fisso di elementi, tutti dello stesso tipo (fig. 10). Non c'è nessun vincolo sul tipo degli elementi, che può essere semplice oppure strutturato. Gli elementi dell'array possono essere selezionati tramite uno o più indici. Il tipo dell'indice può essere un qualsiasi tipo scalare, non necessariamente numerico. Per array con più indici sono legali differenti notazioni, per esempio:

```
array [1..5,1..10] of integer
equivale a
array [1..5] of array [1..10] of integer.

Esempi:
type giorno = (lun,mar,mer,gio,ven,sab,dom);
     bar = (pub,pirata,harry's);
var  tabella : array [giorno,bar] of (aperto,chiuso);
...
...
...
tabella [mar,pirata] := aperto;
if tabella [gio,pub] = chiuso then ...

Cosi', le seguenti:
type mese = array [1..31] of integer;
var  x : mese;

type giorni = 1..31;
     mese = array [giorni] of integer;
var  x : mese;

var  x : array [1..31] of integer;

sono tre dichiarazioni equivalenti. Ancora un caso:

type anno = array [1..12] of mese;
     mese = array [1..31] of giorno;
     giorno = array [1..24] of ora;
     ora = (piacevole,noiosa);
var  x : anno;
x                 denota l' anno x
x [5]             denota il quinto mese dell' anno x
x [5] [7]         denota il settimo giorno del quinto
                  mese dell' anno x
x [5] [7] [13]    denota la tredicesima ora del settimo
                  giorno del quinto mese dell' anno x
```

FIG. 10 - TIPO ARRAY



Record

Un record rappresenta una collezione di un numero fissato di componenti (detti campi) di tipi diversi (fig. 11); ogni campo è denotato da un identificatore, e non possono esserci due campi dello stesso record con il medesimo identificatore.
Esempi:

```
Esempi
type tempo = record
               ora : 0..24;
               minuto,secondo : 0..59
             end;
var x : tempo;
...
...
...
x.ora := 17;
x.minuto := 45;
x.secondo := 33;

type animale = record
                 specie : array [1..20] of char;
                 zampe : integer;
                 domestico : boolean
               end;
```

Record Varianti

Un record può contenere anche una parte variante, che è introdotta dalla parola chiave CASE e consiste in un certo numero di gruppi di campi alternativi, associati a differenti valori del selettore del CASE.

Esempio:

```
type data = record
            mese : (gen,feb,mar,apr,mag,giu,lug,set,
                    ott,nov,dic)
            giorno : 1..31;
            anno : integer
            end;
     origine = (nazionale,estero);
     studente = record
                  nome : array [1..20] of char;
                  datadinascita : data;
                  scuola : array [1..10] of char;
                  provenienza : origine;
                  case origine of
                     nazionale : (citta':array [1..20]
                                    of char);
                     estero : (nazione : array [1..10]
                                    of char;
                               citta' : array [1..20]
                                    of char;
                               " permessosoggiorno:date)
                  end;
```

Set

Un set (fig. 12) rappresenta un insieme di valori costituiti da tutti i sottoinsiemi di un tipo scalare detto tipo di base. Per esempio se consideriamo come tipo di base il seguente:

type ingredienti = (latte, farina, burro, uova, sale, zucchero);

possiamo avere i seguenti sottoinsiemi:

[latte, farina, zucchero],
[burro, uova, sale],
[farina],

che sono tre dei possibili valori del tipo:

type ricetta = set of ingredienti;

Un insieme vuoto può essere denotato da una coppia di parentesi quadre: [ ]

Esempi:

type numeri = set of 0.134;
type legno = set of (quercia, noce, acero, pino, betulla);
type tabella = set of boolean;
type parola = set of char.

File

Un file (fig. 13) rappresenta una sequenza di un numero non fissato di elementi, tutti dello stesso tipo (semplice o strutturato). Gli elementi componenti non possono però essere di tipo file, sono quindi proibiti *file of file.*

Si può accedere ai componenti di un file solo in ordine sequenziale e a un solo componente per volta. La dichiarazione di una variabile F di tipo FILE crea automaticamente una variabile, detta file-buffer, che è dello stesso tipo dei componenti del file, ed è denotata dal simbolo F. Questa variabile non fa parte del file ma, tramite le primitive PUT E GET, permette l'accesso in scrittura e in lettura.

Non essendo possibile l'accesso diretto, esiste un insieme di funzioni e procedure predefinite che operano sul file e realizzano la comunicazione col programma; vediamo quali sono:

Funzione END-OF-FILE: EOF (F)

Ha un risultato di tipo boolean, può essere utilizzata per verificare se è stata raggiunta la fine del file. Se EOF (F) è TRUE allora la variabile F è indefinita, se invece EOF (F) è FALSE allora F contiene l'elemento del file sul quale siamo posizionati. Quando si sta creando un file EOF (F) è sempre TRUE.

Procedura RESET: RESET (F)

Questa procedura apre il file in lettura e posiziona F sul primo elemento, se F è vuoto allora EOF è posta a TRUE e F è indefinita, altrimenti EOF è posta a FALSE e F contiene il valore del primo elemento.

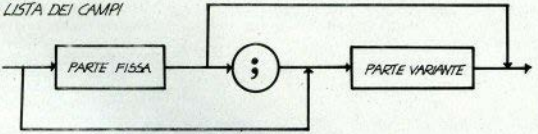
Procedura GET: GET (F)

Permette di posizionarsi sull'elemento successivo di un file dopo averlo aperto in lettura; se tale elemento esiste, il suo valore è accessibile tramite F e EOF è FALSE, altrimenti EOF è TRUE e F è indefinita. Un tentativo di GET quando EOF vale TRUE oppure quando il life non

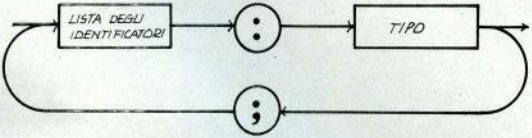
FIG. 11 · RECORD

TIPO RECORD

LISTA DEI CAMPI



PARTE FISSA



PARTE VARIANTE

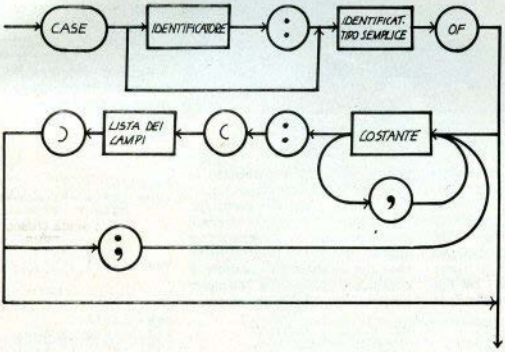
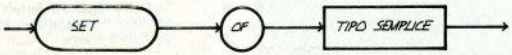


FIG. 12 - TIPO SET



è stato aperto in lettura causa l'uscita dal programma.

Procedura REWRITE (F)

Permette la generazione di un nuovo file e/o l'apertura in scrittura.

Procedura PUT: PUT (F)

Permette di aggiungere un nuovo elemento a un file precedentemente aperto in scrittura; il valore di questo elemento deve essere memorizzato nel file buffer F. Se si cerca di eseguire una PUT senza avere aperto il file si provoca l'uscita dal programma.

Esempio:

```
type mesi= (gen, feb,mar,apr,mag,giu,lug,ago,set,ott,
          nov,dic);
     settimana = (lun,mar,mer,gio,ven,sab,dom);
     appuntamento = record
                    mese : mesi;
                    giorno : settimana;
                    ora : 0..24;
                    minuto : 0..59
                    end;
     agenda = file of appuntamento;
var myagenda : agenda;
    app : appuntamento;
...
...
...
rewrite (myagenda);              ***
myagenda^.mese := apr;             *
myagenda^.giorno := gio;           *  Inserisce un nuovo
myagenda^.ora := 15;               *  elemento nel file.
myagenda^.minuto := 30;            *
put (myagenda);                  ***

reset (myagenda);                ***  Preleva un elemento
app := myagenda^;                  *  dal file, lo assegna
get (myagenda);                  ***  alla var. app e si
                                      posiziona sul
                                      successivo.
```

Tipi Packed

Come si può notare dalla carta sintattica dei tipi, è possibile permettere alla definizione di array, record, set e file la parola chiave PACKED. Questa operazione non influisce sulla semantica della definizione ma serve per risparmiare memoria, compattando i valori del tipo strutturato; lo svantaggio è una perdita di efficienza nell'accesso ai componenti della struttura.

Il prefisso PACKED è particolarmente utile per gli array di caratteri, perchè permette il confronto e l'assegnamento diretto fra stringhe invece che carattere per carattere. È possibile "impaccare" e "disimpaccare" un tipo tramite le procedure PACK e UNPACK.

FIG. 13 - TIPO FILE

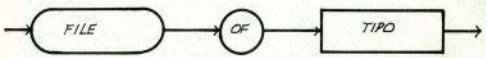
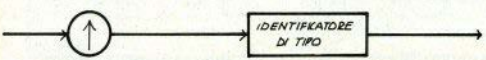


FIG. 14 - TIPO PUNTATORE



Puntatori

Un tipo puntatore rappresenta un insieme di valori che servono a "puntare" ai dati e contengono l'indirizzo dell'elemento puntato. Un puntatore è un meccanismo particolare che si differenzia da quelli visti finora, perchè permette al programmatore di costruire strutture di dati variabili in dimensioni e forma; le componenti di queste strutture possono venire create e distrutte dinamicamente durante l'esecuzione del programma. Supponiamo di avere la seguente definizione di tipo:

type P = T;
var Y : P

dove T è un tipo precedentemente definito; le variabili di tipo P sono quindi dei puntatori a variabili anonime di tipo T. Per operare su queste variabili anonime si utilizzano le procedure predefinite NEW e DISPOSE:

NEW (Y) crea una variabile di tipo T anonima e di valore da definire e ne carica l'indirizzo in Y;

DISPOSE (Y) distrugge la variabile puntata da Y.

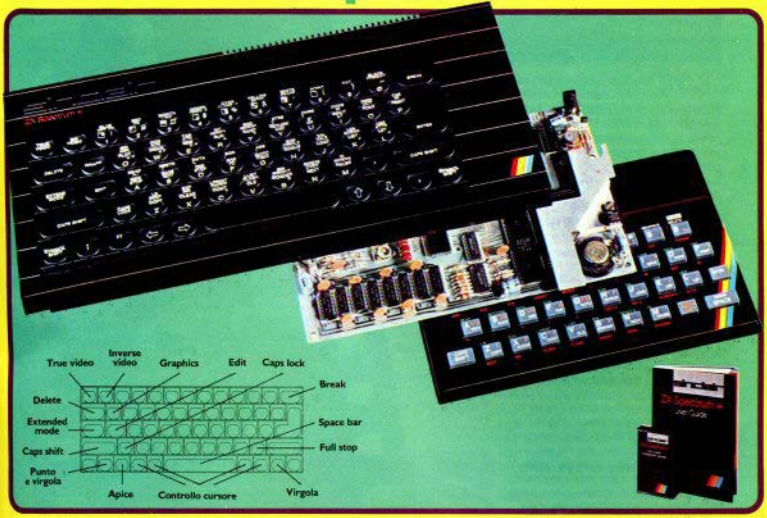
È possibile segnalare che un puntatore non riferisce nessun elemento assegnandogli la costante predefinita NIL.
Esempio:

```
Esempio:
type lista = ^ nodo;
     nodo = record
                info : integer;
                prox : lista
            end;
var list : lista;
.
.
.
new (list);
list^.info := 155;
list^.prox := nil;
```

L'utilità dei puntatori risulterà molto più chiara quando parleremo di ricorsione e della creazione di strutture ricorsive, quali per esempio liste e alberi.

(2 - continua. La prima parte è apparsa sul n. 13)

Trasforma il tuo Spectrum in ZX Spectrum +



Ecco una novità stimolante per i possessori di Spectrum :
Il KIT ORIGINALE SINCLAIR, che promuove lo Spectrum al grado superiore.
Non si richiede vasta esperienza. Basta saper saldare pochi fili.

CARATTERISTICHE:

- Tastiera professionale SINCLAIR con 17 tasti extra.
- Si usa come una normale macchina da scrivere.
- Compatibile con tutto il software e le periferiche Spectrum.
- Completo di una guida di 80 pagine più una cassetta dimostrativa.

a casa vostra subito !!

Descrizione	Q.tà	Prezzo unitario	Prezzo Totale
Kit 48K/Plus		L. 109.000	

Desidero ricevere il materiale indicato nella tabella, a mezzo pacco postale contro assegno, al seguente indirizzo:

Nome
Cognome
Via
Città
Data C.A.P.

SPAZIO RISERVATO ALLE AZIENDE - SI RICHIEDE L'EMISSIONE DI FATTURA
Partita I.V.A.

PAGAMENTO:
A) Anticipato, mediante assegno bancario per l'importo totale dell'ordinazione.
B) Contro assegno, in questo caso, è indispensabile versare un acconto di almeno il 50% dell'importo totale mediante assegno bancario. Il saldo sarà regolato contro assegno.

AGGIUNGERE: L. 5.000 per contributo fisso.
I prezzi sono comprensivi di I.V.A.

DIVIS.
EXELCO Via G. Verdi, 23/25
20095 - CUSANO MILANINO - M...

SI ACCETTANO FOTOCOPIE DI QUESTO MODULO D'ORDINE

THE WILD BUNCH
Spectrum 48K
Firebird

Un gioco di avventura: hanno sparato a un uomo e tu sei accusato del delitto. Per dimostrare la tua innocenza devi rintracciare il vero assassino, sulla base della descrizione data dall'ucciso. Lo sceriffo ha messo un suo agente sulle sue tracce che cercherà di imprigionarti prima che tu possa catturare il membro della "Wild Bunch" che ha commesso il crimine e portarlo al cospetto della Giustizia.

Ma il selvaggio West è terra molto pericolosa: bisogna evitare di essere uccisi nel deserto, mentre si è in cammino tra una città e l'altra, da qualche serpente a sonagli o da qualche pellerossa. E anche nel saloon un baro, un ubriacone o una ballerina possono rappresentare una minaccia mortale se non si è cauti.

The Wild Bunch ha una struttura abbastanza insolita per un'avventura. Per usare un termine preso in prestito dai programmi di archiviazione, potremmo dire che è "Menu driven". Ciò è molto importante: significa che in ogni situazione è il computer a dirci quali sono le azioni che possiamo compiere, e per selezionare una o l'altra opzione è sufficiente la pressione di un tasto o due.

Perciò, anche se il programma è tutto in lingua inglese, non è molto faticoso colloquiare col computer, e non dovrete arrovellarvi a pensare quale è il verbo o il sostantivo adatto per ottenere una certa azione.

In varie fasi del gioco, l'avventura comporta il dover giocare a carte (con grafica a video), oppure si incontrano altri puzzle strategici, cose che rendono questo programma un misto di avventura e di giochini vario ed appassionante.

Pur essendo in basic, i tempi di risposta sono velocissimi e la grafica, quando è prevista, è abbastanza curata.

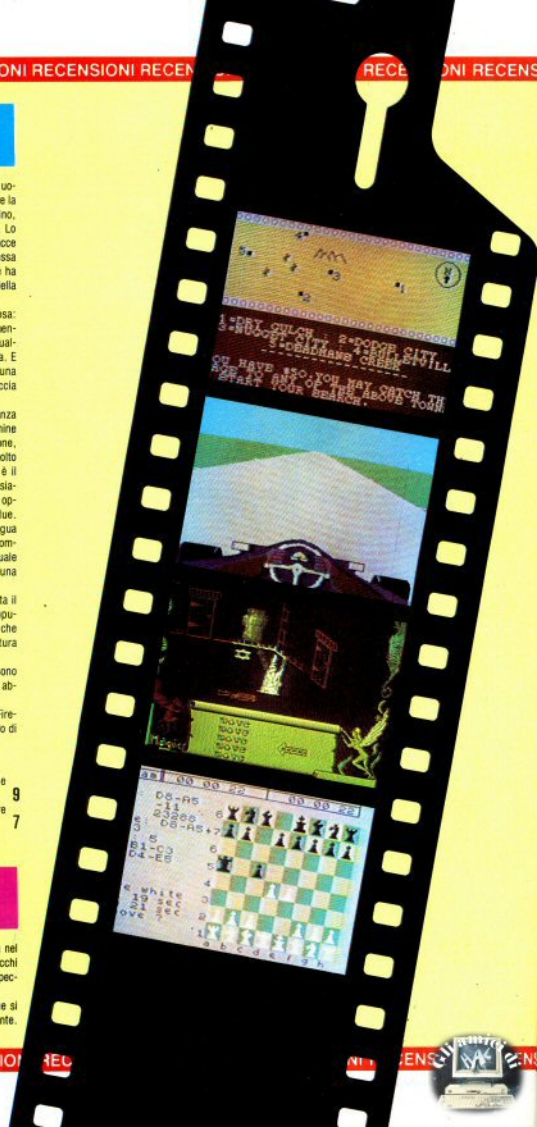
Buona anche la protezione utilizzata dalla Firebird: è assai difficile da superare ed è in grado di mettere fuori gioco il 95% dei duplicatori.

Comandi: avventura "menu driven".
Giocabilità: Ottima data la varietà di situazioni e di falsi di gioco. **9**
Grafica: Quando prevista è buona, per essere generata da basic. **7**

ENDURO
Activision
Spectrum 48K.

La Activision è una casa abbastanza nota nel campo dei videogiochi commerciali per parecchi suoi validi prodotti, ma il software per lo Spectrum è, in generale, molto scadente.

ENDURO è uno dei peggiori programmi che si sia capitato di vedere, e non solo recentemente.



Ne abbiamo deciso la recensione solo per mettere in guardia i lettori da simili incauti acquisti.

Si tratta di una corsa automobilistica, ispirata idealmente al leggendario «Pole Position» della Atari o a «Chequered Flag» della Psion: bisogna superare un certo numero di vetture concorrenti senza scontrarsi e senza andare fuori strada, pena il termine della corsa.

Possiamo soltanto, tramite 4 tasti non ridefinibili, accelerare, frenare, curvare a sinistra o a destra. È utilizzabile un joystick, ma lo sconsigliamo, vista la eccessiva sensibilità dimostrata al più piccolo movimento della leva.

Sul video abbiamo costantemente un indicatore di velocità, uno della fase di gioco in corso e uno che indica quante macchine ancora dobbiamo sorpassare per potere accedere alla successiva fase.

Man mano che si procede non accade nulla di nuovo, solo le macchine avversarie diventano più perfide e ci ostacolano maggiormente quando tentiamo di sorpassarle.

La cosa peggiore di tutto il programma, a parte la totale manzanza di ogni sofisticazione tecnica o fantasia stilistica (tipo: sterzo a 4 tasti, cambio marcia, effetti sonoro, più piste, più tipi di macchina), è senza dubbio la grafica. Se non fosse stato scritto sulle istruzioni d'uso del programma, avremmo pensato che non si tratta di una corsa automobilistica, ma di una sfida tra insetti (cimici?) in un prato; vedere la foto per credere.

Consigliabile a chi, in possesso di uno Spectrum con memoria non espansa, desidera un giochino per fare divertire il fratellino o il figlioletto senza pretese sulla qualità del gioco.

Grafica: certamente veloce, ma ridicola. **2.**

Comandi: semplicissimi ed inconsistenti **4.**

Giocabilità: passa la voglia di giocare solo assistendo al «demo». **3.**

FORMULA 1 SIMULATOR
Spectrum 48K
Mastertronic

Pochi videogiocatori non conoscono lo storico "Pole Position" della Atari, che ha appassionato migliaia di aficionados dei giochi di velocità e simulazione.

Per Spectrum esistono almeno 10 versioni di quel gioco, basato su di una corsa di formula uno in tempo reale, con grafica molto suggestiva, e questo prodotto della Mastertronic è quello che, nel complesso, ci è piaciuto sinora maggiormente.

Sul video abbiamo, oltre alla visione della strada, le indicazioni di: velocità, numero di giri del motore, tempo record della pista, nostro tempo, marcia inserita, giri percorsi.

Inizialmente possiamo scegliere tra una macchina con cambio automatico oppure una normale con 4 marce, che consente uno sfruttamento migliore del motore, ma richiede più abilità, ed è adatta a chi già se la cava bene con volante, freno ed acceleratore.

Una maggiore difficoltà può essere selezionata scegliendo un'opzione che dissemina sulla pista macchie di liquido, che possono farci sbandare o rallentare.

Una delle caratteristiche che rendono molto vario il gioco è la possibilità di scegliere tra 10 percorsi (Silverstone, Monza, Zandvoort...), che sono stati digitalizzati dai programmatori e riprodotti con le loro reali caratteristiche.

È anche possibile esercitarsi prima della competizione senza che vi siano le macchine concorrenti in pista, che invece durante la corsa tentano di metterci fuori strada oppure possono eliminarci se le urtiamo.

La guida si effettua con 6 tasti (4 se la vettura ha cambio automatico) oppure con un joystick (4 marche compatibili) e 2 tasti.

La grafica non è particolarmente sofisticata, ma si è privilegiata il lato tecnico, simulando assai bene la dinamica delle automobili e la geometria dei percorsi. I suoni sono ridotti all'indispensabile.

Comandi: discreti i tasti (non ridefinibili), meglio le manopole.

Giocabilità: elevata dati i 10 percorsi e varie opzioni.

Grafica: non particolarmente rifinita ma tecnicamente valida.

AVALON
Hewson Consultants
Spectrum 48K.

AVALON è una specie di avventura/arcade, secondo una tradizione ormai in voga presso le **software house** inglese, che richiede ragionamento, intuizione, senso dell'orientamento e memoria, oltre alla pura velocità di manovra.

Ci troviamo nel castello del «Signore del Caos», composto da 8 diverse scenografie, tutte differenti tra loro e in grafica tridimensionale, molto originale e suggestiva.

Scopo del gioco è scovare il padrone di casa per ucciderlo, evitapdo le mille insidie ed i vari guardiani disseminati per l'infinità di schermi (stanze) da superare.

Fortunatamente siamo degli aspiranti maghi e, procedendo nell'esplorazione delle stanze, possiamo entrare in possesso di vari poteri magici, che ci consentono di affrontare vittoriosamente molti nemici. Ogni volta che si entra in possesso di un nuovo potere bisogna però imparare a usarlo efficacemente

Anche la capacità di movimento è ottenuta utilizzando un potere magico, e poichè è impossibile usare contemporaneamente due stregonerie, Maroc (il protagonista) è indifeso quando si muove.

A eccezione del potere di movimento, tutte le altre magie sono disponibili per un numero limitato di volte: non è possibile diventare più tre volte invisibile, oppure lanciare per più di tre volte sfere di fuoco contro scheletri o guardie o aguzzini vari.

Procedendo tra una stanza e l'altra si trovano scrigni e bauletti, che bisogna aprire; spesso vi sono contenuti oggetti (chiavi o altro) indispensabili per proseguire il gioco. Similmente agli **adventures** classici, possiamo in ogni momento salvare su nastro la situazione raggiunta, per riprendere senza dovere ripetere dall'inizio.

Grafica: tridimensionale, veloce, originale, suggestiva, colorata. **10.**

Comandi: piuttosto scomodi da usare i poteri. Joystick valido. **7.**

Giocabilità: è molto difficile annoiarsi. Situazione salvabile. **8.**

SUPERCHESS 3.5
Spectrum 48K
CP Software

Senza ombra di dubbio questo programma è attualmente il migliore tra tutti quelli disponibili per Spectrum (ne conosciamo 11) e forse anche tra tutti i programmi scacchistici disponibili per home computer.

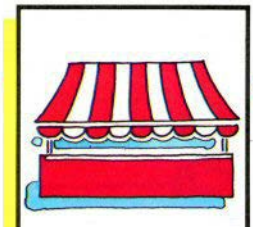
Il suo predecessore, Superchess 3.0, giunse secondo in una gara tra programmi alla quale parteciparono tutti i più noti (Sargon, Mychess, The Turk, Masterchess, Superchess II...) per varie macchine, battuto 3 a 2 soltanto dal Sargon III del Commodore 64. In una prova da noi effettuata questa nuova versione, dello stesso autore, ha stravinto anche col prodigioso Sargon III, concedendo solo 3 patte in 6 partite all'avversario.

Questo programma non dispone, come la maggiore parte dei programmi e delle scacchiere elettroniche, di "livelli" di gioco: si regola l'abilità di gioco assegnando al programma un certo tempo limite, in secondi di riflessione sulla mossa. Questo parametro può essere modificato anche durante il gioco.

Superchess 3.5 conosce un discreto numero di varianti fondamentali del repertorio di aperture classiche (Ruy Lopez, Gambetto di Donna, Difesa Siciliana, Difesa Alekhine...), sia quando gioca col bianco che quando gioca col nero.

Sul video abbiamo: la scacchiera (della quale possiamo regolare tutti i colori), la nostra mossa, la mossa del computer, il numero di mosse fatte (stampabili su carta opzionalmente), gli orologi (ma non esiste la "caduta della bandierina" a tempo scaduto!) e la sequenza di mosse considerata migliore dal computer, che può rappresentare un valido suggerimento per il giocatore inesperto.

Viene anche mostrato il numero di posizioni analizzate, che, sembra incredibile, aumenta di circa 900/1800 al secondo. Ciò significa che il computer, con qualche approssimazione, può rintracciare in pochi attimi le successive due migliori mosse del giocatore avversario e regolarsi di conseguenza. Il manuale è completo e valido.



sinclair réclame

Vendo Scambio Hard

Vendo **Alphacom 32** come nuova, rotoli carta omaggio, L. 170.000. Beppe Fasolis, corso Alba 13, 14100 Asti, 0141/53817.

Vendo **monitor 20" B/N** adattato per Spectrum con ingresso suono, L. 80.000. Roberto Salerno, via Valle Antigorio 10, 20152 Milano, 02/4563547.

Vendo **Spectrum 48K**, pochi mesi, + box acustico, interfaccia joystick programmabile, manuale in italiano, 80 videogiochi, L. 550.000 trattabili. Pietro D'Asta, corso Potenza 170, 10149 Torino, 011/296429 h14-15 o dopo le 20.

Vendo **Spectrum 48K** come nuovo, registratore, programmi di ogni genere, alimentatore, libri, cavi, L. 400.000 trattabili. Marco Maregalli, via Madonna 4, 22063 Cantù (CO), 031/710194.

Vendo **Spectrum 48K** nuovo, imballo, 2 manuali, cassetta dimostrativa Horizons, oltre 100.000 lire di software, L. 360.000; Spectrum Plus 48K stesse condizioni, L. 390.000. Tel. Massimo, ???/3271966 (cena).

Per QL vendo **espansione di memoria 128 k** su scheda Eurocard, con connettore per ulteriori espansioni, tutte le memorie su zoccolo, L. 350.000. Giampietro Sobrero, via Apparizione 15, 16133 Genova, 010/392187.

Vendo **tastiera per Spectrum**, nuovissima mai battuta, L. 50.000. Lanfranco Lanaro, via Leopardi 24, 65100 Pescara, 085/385216.

vendo **Spectrum 48K** issue 3, 6 mesi, alimentatore anti black-out e predisposizione attacco monitor, L. 320.000 (software a richiesta). Giorgio Colombo, via S. Carlo 13, 20035 Lissone (MI), 039/481308.

Vendo **Stampante Seikosha GP 100A** + interfaccia Centronics, L. 450.000; tastiera professionale per Spectrum L. 100.000, tutto in ottimo stato. Florian Ellmenreich, via Dante 78, 39012 Merano (BZ), 0473/31896 mezzogiorno.

Vendo **ZX 81** completo L. 60.000 + espansione 16K L. 80.000, + 2 manuali in italiano. Alessandro Pini, via Zorzi 46, Maranello (MO), 0536/941358.

Vendo **Spectrum 48K + ZX Printer**, come nuovi, software vario, L. 400.000. Tiziano Fiorini, via Trento 11, 52117 Stia (AR), 0575/582142 pasti.

Vendo **Spectrum 48/80K**, + scheda Speedy a richiesta, programmi di tutti i tipi, light pen, "Run" 1-7, 2 libri, ottime condizioni, L. 500.000 trattabili. Francesco Fontana, via Valsesia 44, 20152 Milano, 02/4594807 sera.

Vendo piastra **Rockwell Aim 65 4K** con linguaggio basic e monitor, 5 manuali originali + 1 libro in italiano sul 6502 (Jackson), + stampante termica 20 caratteri Olivetti e stampante di ricambio, + mobile porta piastra in plastica appositamente costruito, uscite seriale e parallela, uscita espansione fino a 64K, prezzo fantastico; oppure scambio con interfaccia 1 + microdrive + stampante 80 colonne. Francesco Ghirotti, via Perilli 11, 48100 Ravenna.

Vendo **Spectrum 80K**, programmi di utilità e giochi, L. 300.000. Marco Prinetti, via Mazzone 14, 13037 Serravalle Sesia (VC), 0163/450127.

Vendo **interfaccia 2** per Spectrum con 2 joystick "Spectravideo" "Triga command", nuovi, imballo originale, giochi in omaggio, L. 110.000. Andrea Bressan, via Europa 2, 35010 Vigodarzere (PD), 049/702656.

Vendo **Spectrum 48K** Natale '84, cavi, usato pochissimo, L. 360.000. Luca Parolari, viale Garibaldi 46B, 30170 Mestre (VE), 041/5059537.

Vendo **Colecovision** consolle modulo turbo super action controllers + 14 cartucce, solo zona Roma. L. 500.000. Giampaolo Moretti, via E.I. Cerva 60, 00143 Roma, 06/5033739 h 18-19.

Vendo computer **Newbrain AD** con display alfanumerico 16 digit, + monitor 12", software vario (fra cui compilatore Pascal), manuale italiano, alimentatore, cavi, registratore, L. 950.000 trattabili. Ivan Moretti, via Mamberetti 40, 20157 Milano, 02/3554126.

Vendo **ZX 81 48K**, alimentatore, cavi, manuale, L. 400.000 trattabili. Marco Galliano, via Fleming 49, 10060 Pinasca (TO), 0121/81719.

Vendo **Spectrum 48K** usato pochissimo, interfaccia e joystick Kempston, manuali inglese e italiano, libro microdrive, circa 30 giochi, L. 499.000 (senza giochi 450.000). Nicola Ponti, via A. Piazzi 2, 21021 Angera (VA), 0331/931096.

Vendo **Spectrum 48K** 1 anno perfette condizioni, imballo originale, + 200.000 lire di software, L. 340.000 anche trattabili. Antonio Colantuono, via Trinità 74, 83100 Avellino, 0825/30945.

Vendo **interfaccia 1 con microdrive** e 7 cartucce, imballati 1 mese, libri e programmi omaggio, L. 300.000. Mauro Donatelli, via Biancardi 4, 20100 Milano, 02/4691535.

Vendo **Spectrum 48K** completo con **presa monitor**, stampante Alphacom 32, tutto poco più di un anno, ottimo stato, utilities fra cui word processor, assemblatore, compilatore ecc., L. 450.000. Angelo Quaglia, via Griffi 6, 21100 Varese, 0332/232470.

Vendo **ZX 81, espansione 16K** in garanzia fino a giugno, 4 cassette programmi, 2 libri, 5 riviste, L. 200.000. Piera Rigolio, via cavour 15, 21013 Gallarate (VA), 0331/796581.

Vendo **Spectrum 16K,** cavi alimentatore, libro italiano, oltre 50 giochi L/M, L. 300.000, o scambio con interfaccia 1 e microdrive. Mariano Marcone, rione 167, isolato K, scala L, n. 245, 80144 Napoli, 081/7014173.

Vendo anche separatamente **Spectrum 48K issue 2, L. 250.000 e tavola grafica Grafpad L. 450.000; utilities e giochi omaggio. Gianluigi Errico, via Porta S. Angelo 1, 05100 Terni, 0744/427106.**

Vendo computer Pet/CBM 4032 con monitor a fosfori verdi (possibilità video a 80 colonne), registratore, manuali originali, joystyck, molti programmi, L. 950.000. Antonio di Gilio, via Monte Cervino 1/9, 30030 Favaro Veneto (VE), 041/611259 h 19-20.

Vendo al miglior offerente **joystick 8 posizioni con 4 microswitch, programmabile, 2 tasti fuoco, + interfaccia 1k compatibile con ogni joystick, cassetta istruzioni in italiano, 2 mesi, usato 3 volte. Enrico Bargelli, via E. Menasse 15, 57100 Livorno, 0586/854785 pasti.**

Vendo ZX 81, alimentatore, cavi,

registratore, video, espansione 16K, manuale originale + "Guida allo ZX 81", ottime condizioni, confezione originale, prezzo da concordare. Vincenzo Musico, via P. Biandino 12, 98100 Messina, 090/2938626.

Vendo **monitor colore RGB cabel 14** mod. MC 3700 nuovo in garanzia, L. 540.000; BCPL per Sinclair, L. 40.000. Roberto Mariconda, via Girardini 16, 3100 Udine, 0432/204033 h 14,30-15,30.

Vendo **Spectrum** come nuovo, cavi, manuali italiano e inglese, ZX Printer, registratore, libro sul L/M, L. 400.000. Roberto Tirone, via Pacini 76, 20131 Milano, 02/295064.

Vendo **Spectrum 48K,** cavi, alimentatore, interfaccia e joystick Kempston, interfaccia monitor, manuali, 250 programmi, libri, L. 500.000. lorenzo De Cola, via Saffi 60, 47042 Cesenatico (FO), 0547/81152.

Vendo **Spectrum 48K,** cavi, alimentatore, imballo e manuale originali, oltre 120 programmi e utilities, L. 370.000. Alberto Burro, viale Venezia 7, 37100 Verona, 045/522273.

Compro Cerco Varie

Si è costituito il **QL User Club Pavia.** Per informazioni: QL UC PV, c/o Paolo Carnevale, via Cadisana 2, Zerbolo (PV).

Cerco **stampante, penne ottica, interfaccia joystick** e altro hardware in cambio di software Spectrum 16-48K. Mariano Marcone, rione 167 isolato K scala L n. 245, 80144 Napoli, 081/7014173.

Effettuato **traduzioni dall'inglese** di testi manuali e istruzioni inerenti i computer Sinclair, prezzi modici o in cambio di software 48K; per lista traduzioni pregasi allegare francobollo. Rossano Mariotti, via E. Curiel 7, 61032 Fano (PS).

Cerco **istruzioni in italiano** per il programma Masterfile. Fabrizio Martano, via Don Sturzo 7, 58100 Grosseto, 0564/492806 pasti.

Cerco programmi di **simulazione, speedloading,** duplicatori hearderless. Flavio Chianese, via Virgilio 17, 34170 Gorizia, tel. 33183.

Cerco possessori Spectrum **zona Avellino.** Luciano De Lisa, via Bellabona 109, 83100 Avellino, 0825/37977 - 22950.

Cerco **stampante per Spectrum 80 o + colonne** a buon prezzo. Tiziano Fiorini, via Trento 1 x, 52017 Stia (AR), 0575/582142 pasti.

Cerco **ZX Printer per ZX 81** e penna ottica, in buon stato, tutto e meno di L. 100.000. Andrea Belvini, viale Oberdan 35, 31100 Treviso, 0422/260316.

Cerco **gli adventures:** "Arrow of Death" (1 e 2), "Message from Andromeda", "Ten little Indians". Piermarco Gordini, via Valbondione 111, 00188 Roma, 06/6422924 sera.

Cerco il libro inglese sul **disassemblato della ROM** anche in fotocopia. Paolo Cortelli, via Calvaret 65, 40129 Bologna, 051/367893.

Cerco **possessori Sinclair QL** per scambio programmi, libri e riviste. Roberto Corbetta, via Parini 42, 21047 Saronno (VA).

Cerco i numeri da 1 a 6 di **Sinclair Computer** in cambio di software. Luca Cavallari, via ?? 29, 21142 Milano, 02/8262007 dopo le 19.

Vendo per Spectrum **traduzione dattiloscritta manuale;** Enciclopedia di Elettronica e informatica Jackson, 8 volumi completi e rilegati, L. 180.000. Giovanni Natale, viale Trieste 36, 93100 Caltanisetta, 0934/51411 pasti.

Cerchiamo **utenti Spectrum** in tutta Italia per il club "Spectromania"; iscrizione L. 2000, giornalino mensile, sconto 50% sul software. Spectromania Club, via delle Romite 8, 50124 Firenze/Galluzzo, 055/2048905.

Scambio **informazioni sugli adventures** Sandro Pase, via R. Balestra 1, 00152 Roma, 06/5386647.

Cerco **assemblatore / disassemblatore** e altre utilities in L/M con manuale per ZX 81. Mario Rotigni, via Piave 25, 24052 Azzano San Paolo (BG).

Per possessori Spectrum: si è costituito in Bari il primo **Spectrumclub**. Rivolgersi a: Paolo Dadone, via Stradella del caffè 8/A, 70124 Bari, 080/414398.

Cerco stampante **Alphacom o Seikosha,** per scambio con software non protetto. Stefano Flattesi, via Marche 28, 60019 Senigallia (AN), 071/6621155.

Vendo i libri: «Alla scoperta dello ZX Spectrum», «Programmazione dello ZX Spectrum», «77 programmi per Spectrum» a l. 14.000 e «Giochiamo con lo Spectrum» a L. 8.000. Giulio di Giuliomaria, via Campo de' Fiori 19, 00186 Roma, 06/6564632.

Si costruiscono **periferiche per ZX 81** o se ne forniscono progetti. Club Electra, c/o L. Bettero, via Brioschi 7, 20091 Bresso (MI).

Compro **rotoli per stampante ZX Printer.** Luigi Caselli, via Alcuino 5, 20149 Milano, 02/383247.

Cerco **programmi scientifici** per Spectrum e adventures, possibilmente grafici, piantine e consigli; cerco numeri di Sinclair Computer 1-6, anche in fotocopia. Francesco Ghirotti, via Perilli 11, 48100 Ravenna, 0544/422254.

Compro **ZX 81** a prezzo conveniente; vendo annata completa 1984 della rivista "Personal software". giuseppe Cardella, via Martogna 46, 91100 Trapani, 0923/48454.

Cerco **per Spectrum 48K i seguenti programmi: Daleyt, Decathon, Match Play, BMX Racers, Knight Lore, Cyclone Booty, Pyjamarama, Select One, in cambio di altro software; solo zona Roma. Alvaro di Giuseppe, viale di Porto 263, 000057 Macarese (ROMA), tel. 6961225.**

Cedo 400 programmi altissima risoluzione grafica in cambio di 2 floppy disk drive per spectrum + un disco. Franco Tortoli, via delle Romite 8, 50124 Firenze, 055/2048905.

Cerco **sprotettori e duplicatori per Spectrum.** Paolo Della Capanna, via Fratti 764, 55049 Viareggio (LU), 0584/50309.

Vendo **"Scienza e vita",** 6-12 1981, 1982-83-84 complete. L. 85.000 escluse spese di spedizione. Luca Marmo, via privata Peiranze 17, 18038 Sanremo (IM), 0184/881137.

Cerco **interfaccia joystick** Kemston funzionante max. L. 20.000. Dario Carraro, via IV novembre 33, 30010 Campagna Lupia (VE), 041/460012.

Cerco **per QL programmi** su microdrive o listati da riviste inglesi. Raimondo Zagami, via Del Plutino 37, 89100 Reggio Calabria, 0965/330377.

Cerco mappa **istruzioni Atic Atac.** Stefano Moro, via Di Vittorio 76, 20097 S. Donato Milanese (MI), tel. 5272518.

Cerco i programmi **Decathlon** della Ocean e il **Pascal-compiler della Hi-soft per Spectrum, in cambio di altro software. Claudio Falcone, via A. Manzoni 16, 30030 Salzano (VE), 041/437757.**

Cerco HI-soft Pascal 1.5 con manuale completo di "turtle" e suono, registratore e/o microdrive compatibile; possibilmente in zona, ZX microdrive e ZX interface 1, solo se affare; sono disponibile per scambio di idee hard e soft di ogni genere. Luca Lombardo, via Don Bosco 92, 18019 Vallecrosia (IM).

Vendo Scambio Soft

Vendono / scambiano software per Spectrum (dove non diversamente specificato)

Zona Caserta. Sergio Triolo, via Sturzo 1, 81020 S. Nicola La Strada (CE)

Zona Brescia. Michele Lombardi, via Vivaldi 14, 25123 Brescia,

Flavio Chianese, viale Virgilio 17, 34170 Gorizia, tel. 33183.

Per ZX 81. Vendo software originale Psion. Luigi Caselli, via Alcuino 5, 20149 Milano, 02/383247.

Per ZX 81 e Spectrum. Fabrizio Martano, via Don Sturzo, 7, 58100 Grosseto, 0564/492806 pasti.

Luciano De Lisa, via Bellabona 109, 83100 Avellino, 0825/37977-29985.

Per Spectrum e Ql. Gabriele Gargini, piazza della Repubblica 56, 56036 Palaia (PI), tel. 822377.

Per QL in L/M. Gianluca Mercuri, via Pigafetta 84, 00154 Roma, 06/5740969.

Zona Torino. Giovanni Ossola, via Tollegno 39/E, 011/279397.

Daniele Buzza, via Bella Vista 24-26, 09134 Pirri (CA), tel. 500623 dopo le 20.

Davide Cantoni, viale Mentana 19, 43100 Parma, 0521/73968.

Vendo programma totocalcio 48k, velocissimo, per elaborazione complessa sistemi ridotti, L. 20.000 comprese cassetta, istruzioni e spedizione. Maurizio Leone, via Gaio Melisso 16, 00175 Roma, 06/7662671.

Per QL. Roberto Ghezzi, via Volontari del sangue 202, 20099 Sesto S. Giovanni (MI), 02/2485511.

Per QL. QL User Club, via Ducdo 3, 50047 Prato (FI).

Francesco Calvelli, via Paisiello 8, 50018 Scandicci (FI), 055/753544.

Ezio Pavone, via Regina Margherita 73, 95024 Acireale (CT), 095/601931.

Giuseppe Armanni, via Campofiore 44, 50136 Firenze, 055/678472.

Beppe Fasolis, corso Alba 13,14100 Asti, 0141/53817.

Alvaro Di Giuseppe, viale di Porto 263, 00057 Maccarese (Roma),tel. 6461225.

Per Spectrum e QL. Gianfranco Baliello, casella postale 52, 30100 Venezia, tel. 28740.

Pierangelo Pensa, via Buonarroti 32/64, 22036 Erba (CO), 031/640574.

Lamberto Righetti, via della Vittoria 29, 19036 Santerenzo (SP). Inviare bollo per lista.

Daniele Di Giovanni, via R. Fucini 112, 00137 Roma, 06/8277222.

Simone Frosini, via Cesaipino 20, 54100 Arezzo, 0575/353393.

Per ZX81. Alberto Severini, via Oleandri 104, 60019 Senigallia (AN), 071/64373.

Massimo Baiano, via Battisti trav. privata 11, 80059 Torre del Greco (NA).

Programmi per Spectrum, elenco su cassetta + 2 programmi omaggio, L. 10.000. Guido Gardinali, via Borgonovo 35, 27038 Robbio (PV).

Vincenzo Emerilli, via Monfalcone 41, 95033 Biancavilla (CT).

Zona Piacenza. Vendo Extgended Basic per Spectrum 48k, L. 10.000 cassetta esclusa. Edo Mara, via Gonzaga 22, 29100 Piacenza, 0523/71467.

Bruno Ballarini, via Magenta 5, 10012 Bollengo (TO).

Giochi, utilities, ottimo programma di statistica. Angelo Fiorillo, Via Cerreto Di Spoleto 24, Roma, 067883256 dopo le 22.

Giochi, utilities, un copiatore microdrive-microdrive o tapemicrodrive. Andrea Dell'Era, via Pio Emanuelli 55, 00143 Roma, 06/5034279 pasti.

Per QL. Marco Fattorini, via L. Viani 21, 50142 Firenze, 055/711629.

Per QL Andrea Galli, via Palagetta 212, 50017 S. Piero a Ponti.

Giorgio Brunetti, fraz. Morro 54, 06049 Spoleto (PG), tel. 45190.

Antonio Caputo, via Madonna della Mercede 1, 98100 Messina, tel. 773628.

Eraldo Taioli, via F. Braganti 8, 47100 Forlì, 0543/65633.

4 programmi in una cassetta per fare qualsiasi sistema al totocalcio. Marco Tronci, via G.C. Cordara, 00179 Roma, 06/7827234 pomeriggio.

Francesco Ventura, via Lettini 1, 70059 Trani (BA), 0883/47339 pasti.

Alessandro Poletti, via G. Lucetti 2, 54031 Avenza (MS).

Marco Arzani, via Giovanni XXIII 14, fiorenzuola d'Arda (PC), 0523/982401.

Giacomo Spagnoli, via Molise 16, 57100 Livorno, tel. 852059.

Per ZX 81. Club Electra, c/o L. Bettero, via Brioschi 7, 20091 Bresso (MI).

Sergio Zardo, via 4 Novembre 24/A, 21049 Uboldo (VA), tel. 9639929.

Zona Roma. Piermarco Gordini, via Valbondione 111, 00188 Roma, 06/6422924 sera.

Andrea Musetti, via Carlo Sforza 7, 54031 Avenza (MS)

Roberto Paolini, via I maggio 6 b, 52040 Terontola (AR), tel. 67208.

Per QL Roberto Corbetta, via Parini 42, 21047 Saronno (VA)

Giovanni Natale, viale Trieste 36, 93100 Caltanisetta, 0934/22775-51411 pasti.

(segue da pag. 43)

Simulatori

gioco.

La strumentazione occupante il terzo inferiore dello schermo, ben rifinita graficamente, comprende: tachimetro, altimetro, orizzonte artificiale, visore d'assetto, VSI (velocità verticale), radar d'atterraggio o di caccia (con collimatori), posizione carrello, potenza motori, scorta carburante.

Certamente la cosa piú avvincente di questo programma è la possibilità di simulare dei veri e propri combattimenti aerei ad alta quota, contro altri caccia (necessariamente con grafica molto stilizzata sul video), che minacciano di bombardare le nostre basi di appoggio. Come (dicono) avvenga su un vero F15, è possibile effettuare acrobazie aeree ad alta velocità, e presto ci si ritrova a compiere i vari "loop", "tonneau", scivolate d'ala e voli rovesciati per sorprendere i nemici.

Certo, durante le "battaglie" il programma cessa praticamente di essere un simulatore realistico, per assomigliare piú a un arcade game, ma il fascino è indiscutibilmente maggiore.

Delta Wing. (Creative Spark Thorn EMI). È di recentissima produzione (1984), e ne esiste anche una versione adatta a funzionare con due Spectrum muniti di Interface 1 collegate, permettendo a due "avversari" di giocare contemporaneamente.

Certamente questo è il meno "serio" dei tre simulatori presentati, tanto è vero che non vengono neppure precisate le caratteristiche tecniche dell'aereo a cui i programmatori si sono ispirati. Vi sono tre livelli di difficoltà e scopo immutabile del gioco è di bombardare le basi aeree nemiche e proteggere le nostre, avendo come avversari dei guizzanti caccia supersonici.

I controlli a disposizione del pilota sono: cloche, flaps, throttle, mappa, portata radar, effetti sonori on/off (se collegata la "sound-box" della Fuller), fuoco cannoncini e sgancio bombe. Gli strumenti visibili in gradevole grafica sono: altimetro, tachimetro, VSI, potenza motori, carburante, segnalatore di stallo, carrello, orizzonte artificiale, posizione radar, indicatore bombe, flap.

La grafica è velocissima, tanto da far assomigliare il programma decisamente piú a un arcade che a un simulatore realistico. Sullo schermo sono visibili la mano del pilota e le gambe, che si muovono secondo i comandi impartiti. Le istruzioni sono molto belle ed estese, pur essendo il programma poco complesso.

Per quanto ci risulta tutti questi programmi sono regolarmente importati, e hanno prezzi compresi tra le 9.000 (versione italiana di Flight Simulation della Mantra Software) e le 30.000 lire.

Per concludere, elenchiamo gli altri simulatori di volo per Spectrum a noi noti: Combat Lynx (*Durell*-elicotteri), Nighflite I & II (*Hewson*), Glider (*Sinclair*-alianti), Tomahawk (*Digital Integration* -Elicotteri), Heathrow (*Hewson*-controllo da terra), F15 (*Us gold*).

Tasti (segue da pag. 44)

SYMBOL SHIFT

Segno di sottrazione. Questo operatore aritmetico non dovrebbe avere problemi per nessuno. Ha priorità uguale all'addizione, inferiore a moltiplicazione e divisione, superiore agli operatori logici.

CAPS SHIFT

"J" maiuscola

MODO "E"

Istruzione VAL. Legge una stringa come espressione numerica, assegnandola a una variabile o stampandola.

Per poter essere letta come numero, la stringa deve essere costituita da caratteri numerici, in formato e dimensione accettati dal computer, oppure corrispondere al nome di una variabile che in quel momento risulti inizializzata.

Gli usi di VAL sono molteplici. Uno dei più frequenti, per fare un esempio, ha lo scopo di risparmiare memoria: in programmi molto lunghi, con molte variabili con valori piccoli, la loro assegnazione a stringhe lette con VAL consente un notevole guadagno di bytes (perdendo un po' in velocità); lo Spectrum infatti non dispone dell'opzione INTEGER, per lavorare con numeri interi, perciò tutte le variabili numeriche vengono rappresentate in formato floating point, con un ingombro di 5 bytes ciascuna. Cfr. anche SC n. 7 pag. 25.

SYMBOL S. IN "E"

Istruzione VAL$. Daremo un premio a chi ci manda un listato con una situazione in cui non si possa fare a meno di questa istruzione: dobbiamo confessare che non riusciamo a comprendere l'utilità di leggere una stringa all'interno di un'altra stringa (dev'essere quindi dentro tre doppi apici, o un numero dispari superiore a tre), per ricavarne una stringa semplice.

K

NORMALE

Comando LIST. Produce sul video la stampa del programma basic attualmente in memoria. Senza argomenti, il listato inizia dalla prima linea: specificando un numero (o una variabile numerica inizializzata), vedremo il programma a partire dalla linea con questo numero o, se non esiste, dalla prima con un numero maggiore. Se non esiste nemmeno questa, si ha semplicemente un O.K.

L'operazione di LIST, analogamente a PRINT, presuppone sempre l'invio a un canale logico: quando non viene specificato, dando cioè semplicemente LIST, si sottintende trattarsi dal 2 (parte alta dello schermo); negli altri casi deve essere esplicito. Per esempio il 3 (preceduto dal solito cancelletto, SYMBOL SHIFT + "3") invia il listato alla stampante, lo 0 alla parte bassa dello schermo.

Con l'interfaccia 1, i canali logici da 4 a 15, debitamente FORMATted & OPENed, possono venire assegnati a microdrive, porta seriale o rete locale, e il LIST invitato a una di queste periferiche.

In questo stesso numero di SC trovate un programma per un LIST più versatile.

SYMBOL SHIFT

Segno di addizione. Nemmeno questo dovrebbe crearvi problemi, dato che lo si usa nello stesso modo che ci è stato insegnato alle scuole elementari. La priorità è la stessa del segno di sottrazione.

CAPS SHIFT

"K" maiuscola.

MODO "E"

Istruzione LEN. Legge la lunghezza in caratteri di una stringa (da LENgth = lunghezza):

```
PRINT LEN a$
LET w = LEN "tamarillobrillo"
```

Il primo esempio stampa un numero corrispondente alla lunghezza della variabile *a$* (che ovviamente deve esistere), il secondo assegna a w il valore 15.

La funzione LEN trova applicazione soprattutto nel trattamento delle stringhe, quando occorre incolonnare, allineare, stampare in tabelle ordinate, etc.: tutto quello che si fa in altri basic con PRINT USING.

Se nelle stringhe che volete elaborare avete inserito attributi di colore (inchiostro, carta, lampeggio, etc., che si assegnano con la prima fila di tasti in modo "E" - cfr. SC n. 9/10), questi vengono conteggiati da LEN, in misura di due caratteri per ogni codice inserito, anche nella stampa (sia sul video che su carta) non occupano nessun spazio. La stringa dell'esempio, con aggiunti all'inizio gli attributi per luminosità e lampeggio e il riporto alla normalità alla fine, risulta essere lunga 23 caratteri anzichè 15.

SYMBOL S. IN "E"

Istruzione SCREEN$. Ha due usi, ben distinti tra loro. Associata alle funzioni di colloquio con l'esterno (LOAD, SAVE, VERIFY) serve per il caricamento, dal registratore o da altra periferica, di una "schermata". L'istruzione

```
LOAD "nome" SCREEN$
```

È una facilitazione d'uso per

```
LOAD "nome" CODE 16384, 6912
```

e carica nell'area di memoria destinata al display-file un blocco di bytes, teoricamente qualsiasi.

Infatti in qualche caso la memoria di schermo viene usata per sistemare una routine in linguaggio macchina (cfr. in questo numero di SC il programma "Bubble-sort in ling. macchina"). Ma lo scopo fondamentale, come ben sapete, è il caricamento delle schermate di presentazione dei programmi.

Il secondo uso, nel formato

```
PRINT SCREEN$ (x,y)
LET a$ = SCREEN$ (x,y)
etc.
```

ritorna una stringa di un carattere, che corrisponde a quanto trova stampato sullo schermo nella posizione specificata da x,y. Se ciò non corrisponde a un carattere del set di cui è provvisto lo Spectrum (per esempio, se si tratta di parte di un disegno ottenuto con PLOT), ritorna una stringa nulla.

L

NORMALE

Comando LET. Assegna a una variabile alfanumerica il valore di un'espressione. Nel formato generale

LET a = espr

a può essere il nome di una variabile sia numerica che stringa, con o senza indici (in questo caso sarà preceduta da un enunciato DIM); *espr* può essere qualsiasi espressione omogenea con quanto sta a sinistra dell'uguaglianza (cioè o numeri, o stringhe).

SYMBOL SHIFT

Segno di relazione "uguale". Sul suo significato e uso nelle operazioni algebriche dovreste sapere già tutto; vale invece la pena di ricordare che può avere il ruolo di operatore di confronto, al pari dei simboli indicanti minore, maggiore o diverso, sia per numeri che per stringhe. Cfr. Questa rubrica sui nn. 11/12: troverete molti esempi d'uso alla trattazione degli operatori relazionali e logici (AND, OR).

CAPS SHIFT

"L" maiuscola.

MODO "E"

Comando USR. Eccoci alla famigerata istruzione che "chiama" il linguaggio macchina *(User Subroutine Request)*.

USR richiede come argomento l'indirizzo di partenza, cioè la locazione di memoria in cui si trova la prima istruzione che deve essere eseguita, e ritorna il contenuto della coppia di registri BC.

Normalmente viene associata a RANDOMIZE, in modo che al rientro dal linguaggio macchina l'unico effetto sia di reinizializzare il generatore di numeri causali. Se questo va evitato, o se sis ha bisogno del contenuto di BC, si possono usare altre forme, come

```
PRINT USR xxx
LET q = USR xxx
etc
```

C'è anche un altro uso di USR, ed è la costruzione dei caratteri grafici definiti dall'utente, noti più brevemente come UDG. Il metodo è già stato descritto più volte: rivediamolo brevemente. Su una matrice di 8x8 quadretti si disegna il carattere desiderato; leggendo la matrice per file orizzontali, si ricavano otto numeri binari, assegnando o ai quadretti rimasti bianchi e 1 a quelli da riempire; facoltativamente, si possono trasformare in decimali con PRINT BIN; ora si esegue questo ciclo:

```
FOR x = 0 TO 7
READ n
POKE USR "a" + x,n
NEXT x
DATA 1,2,3,4,5,6,7,8
```

dove nella linea DATA si trovano i numeri relativi alle otto file di quadretti, dall'alto in basso; "a", che può essere scritto usando indifferentemente la lettera minuscola, maiuscola o il modo "G", costruisce il carattere UDG corrispondente al tasto A, per i successivi si sostituiranno le altre lettere dell'alfabeto.

SYMBOL S. IN "E"

La funzione ATTR ha la forma

ATTR (x,y)

in cui gli argomenti indicano rispettivamente la linea e la colonna di una posizione dello schermo; hanno quindi range utili da 0 a 31 e da 0 a 21, come per AT.

Il numero restituito dipende dallo stato di quattro variabili, relative ai cosiddetti 'attributi' di quella posizione, e cioè:

FLASH - 0 inattivo, 128 attivo
BRIGHT - 0 inattivo, 64 attivo
PAPER - 8 moltiplicato il codice
INK - codice

La somma dei quattro numeri è il risultato dell'esecuzione di ATTR. L'interpretazione di tale risultato è utilizzabile per analizzare una situazione, per esempio le possibilità di movimento all'interno di un labirinto.

ENTER

È senza dubbio il tasto più premuto di qualsiasi tastiera. Serve, genericamente, per eseguire l'operazione scritta sulla parte bassa dello schermo, nell'area di edit.

Se non c'è niente che attende di essere eseguito, si provoca solo un LIST automatico; se non ci sono programmi da listare, non succede niente.

Se in edit-area avete uno o più comandi, premendo ENTER il computer prova a eseguirli; se la sintassi non è corretta, appare un secondo cursore lampeggiante, e il computer resta in attesa della correzione.

Se le istruzioni sono precedute da un numero positivo inferiore a 9999, vengono riportate nella parte alta dello schermo, diventando così una linea di programma; con una sintassi sbagliata si ha lo stesso effetto del caso precedente.

Se un programma sta girando e si trova in attesa di INPUT (cfr.), la pressione di ENTER lo fa ripartire: per un imput numerico occorre qualcosa (sta poi al programma verificarne la validità), l'input alfabetico accetta anche la stringa nulla.

Se il cursore si trova in modo "E" o "G", la prima pressione lo mette in stato "L" (o "K"); una seconda pressione riconduce a uno dei casi precedenti.

DIGITARE STANCA

DIGITARE STANCA!
I programmi più interessanti spesso sono molto lunghi, un listato pubblicato è faticoso da leggere...
Sinclair Computer vi offre un'alternativa: le cassette con tutti i programmi pubblicati sulla rivista.
Ogni nastro contiene il software di un numero di Sinclair Computer, a un prezzo incredibilmente basso: solo 5.800 lire (+ 1.000 lire per spese di spedizione).
Riceverete le cassette direttamente a casa vostra, utilizzando il coupon qui a fianco.

DIGITARE STANCA è un'iniziativa

Systems

Desidero ricevere le cassette con il software pubblicato sui seguenti numeri di Sinclair Computer:

.....

importo L. .....
spese di spedizione L. 1.000

Totale L. .....

☐ ho versato l'importo sul c/c postale n. 37952207 (allego fotocopia della ricevuta di versamento)
☐ acclude assegno non trasf.
n. ..... (banca .....)
intestato a SYSTEMS Editoriale V.le Famagosta 75, 20142 MILANO

nome .....

cognome .....

via .....

CAP/città .....

Ritagliare e spedire in busta: a Systems Editoriale v.le Famagosta 75, 20142 Milano.

ABBONATEVI A SINCLAIR COMPUTER

Utilizzate i tagliandi per abbonarvi, collaborare, chiedere o darci consigli, pubblicare un'inserzione per comprare, vendere, cercare contatti.

Ritagliate e spedite in busta chiusa a:
Sinclair Computer, viale Famagosta 75, 20142 Milano.

ABBONAMENTO

Registrate il mio abbonamento annuale a:
☐ Sinclair Computer (L. 28.000) ☐ Computer + Sinclair Computer (L. 55.000)
☐ Ho versato l'importo sul c/c postale n. 37952207
☐ Accludo assegno non trasferibile n. ..... banca .....
intestato a SYSTEMS Editoriale, v.le Famagosta 75, 20142 Milano
Il mio computer é ☐ ZX81 ☐ Spectrum ☐ .....

possiedo	SI	NO
stampante	☐	☐
microdrive	☐	☐
joystick	☐	☐
	☐	☐

Nome .....
Via ..... n. .....
CAP. ☐☐☐☐☐ città .....
Tel. ..... prov. .....

N.B. L'abbonamento è annuale (11 numeri) e decorre dal primo numero edito dopo il ricevimento della sottoscrizione.

COLLABORAZIONE

Desidero collaborare a Sinclair Computer
Invio il programma " ..... "
listato e registrato su cassetta, con un articolo di commento.
Garantisco che il software è originale e vi autorizzo a pubblicarlo.
Per il compenso scrivetemi al seguente indirizzo:
Nome .....
Via ..... n. .....
CAP. ☐☐☐☐☐ , città ..... prov. .....
Tel. .....
N.B. Il materiale anche non pubblicato non viene restituito

HELP

Nome .....
Via ..... n. .....
CAP. ☐☐☐☐☐ città .....
Tel. ..... prov. .....

CLASSIFICA

Questo mese ho acquistato / provato i seguenti programmi
e li valuto così (max tre titoli):

TITOLO	Ottimo	Buono	Mediocre	Deludente

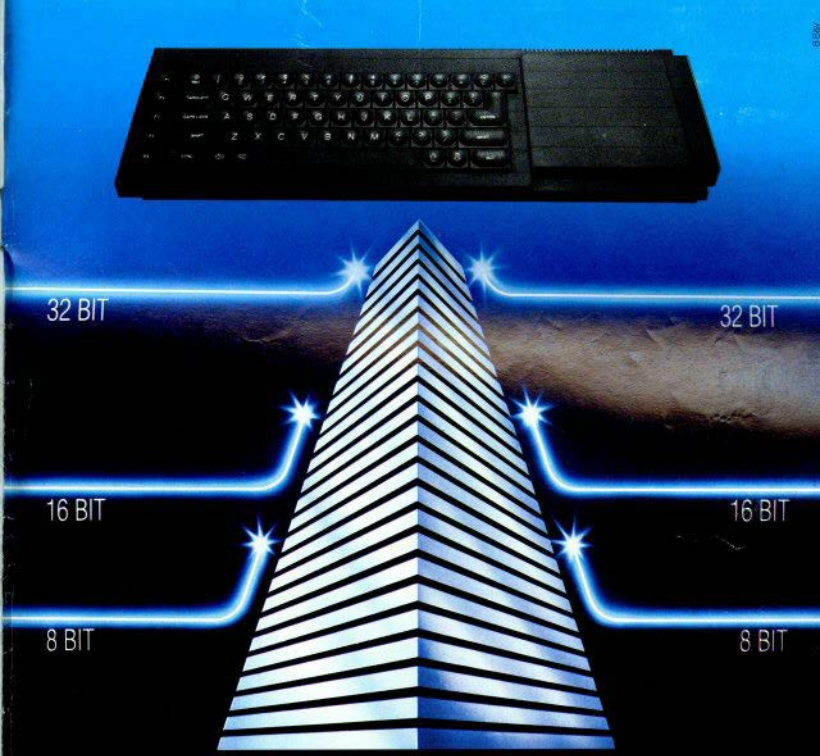
nome e indirizzo (facoltativo) .....

INSERZIONI

☐ VENDO
☐ COMPRO
☐

Nome .....
Via ..... n. .....
CAP. ☐☐☐☐☐ città .....
Tel. ..... prov. .....

Avvertenze:
ABBONAMENTI: scrivete l'indirizzo completo.
COLLABORAZIONE: il listato non è indispensabile, la cassetta sì.
HELP: non accludete francobolli, non rispondiamo privatamente
INSERZIONI: la rubrica è destinata agli scambi tra privati; la redazione si riserva il diritto di modificare o cestinare gli annunci palesemente speculativi.



SINCLAIR QL:
AL VERTICE DELLA NUOVA GENERAZIONE

Sinclair QL rivoluziona il mondo dei computer, perché combina le dimensioni di un home con la potenza e le capacità di un mini.
QL è l'unico computer, nella sua fascia, ad impiegare il microprocessore a 32 bit, quando gli altri si fermano a 8 oppure 16.
La sua portentosa memoria è di 128 KRAM espandibile a 640.
I quattro programmi applicativi, già incorporati, sono immediatamente utilizzabili e superano, in qualità, il software dei microcomputer esistenti.

Ha la possibilità di multitask e può essere inserito in reti di comunicazione.
Grazie ai due microdrive e al software incorporati, Sinclair QL, nella sua confezione originale, è già pronto per l'uso: basta collegarlo ad un video.
E pensare che tutta questa tecnologia pesa meno di due chili e trova spazio in una normale 24 ore.
Un computer così non poteva che essere Sinclair.

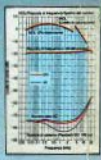
sinclair

Distribuzione esclusiva: GBC Divisione Rebit.

Tutti i prodotti Sinclair, distribuiti da GBC Divisione Rebit, sono corredati da regolare certificato di garanzia ita

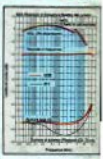
Le nuove Maxell

UDI



Vanta una tecnologia senza confronti nella sua categoria. Infatti tutte le proprietà magnetiche del suo nastro sono esaltate al massimo grazie alla nuova particella magnetica "Ferricrystal" che, a differenza di quelle tradizionali, è **assolutamente non porosa**. La UDI, quindi, offre un livello d'uscita migliorato su tutta la gamma di frequenze, specialmente nelle medie e basse; una più ampia gamma dinamica, ed ottime caratteristiche di lownoise. Adotta la meccanica P.A. (PHASE ACCURACY) per garantire la massima stabilità di svolgimento del nastro e per contenere la differenza di fase tra i canali stereo entro i 10°. Può essere usata su qualsiasi tipo di registratore.

UDII



È la soluzione ideale per chi vuole una cassetta di **categoria superiore** ad un **costo contenuto**. Utilizza un nastro di posizione "Chromo" (CrO_2) prodotto con la stessa tecnologia dei nastri XL e XL-S Maxell.

Infatti il nastro della UDII è composto dalle collaudate particelle magnetiche "Fine Epitaxial", ulteriormente perfezionate. Di conseguenza è aumentata tutta la **gamma dinamica** del nastro, il livello d'uscita alle **medie** ed **alte** frequenze, mentre il livello di rumore di bias ed il rumore di modulazione sono ridotti praticamente a **zero**.

Il meccanismo di scorrimento P.A. e la perfezione dei gusci della cassetta assicurano un regolare svolgimento del nastro, mantenendolo sempre perpendicolare alla testina del registratore (differenza di fase entro i 10°).

maxell

È TUTTA UN'ALTRA MUSICA